MSA와 DevOps

(Microservice Architecture & DevOps)

2019. 12.

안 종 석

james@jslab.kr

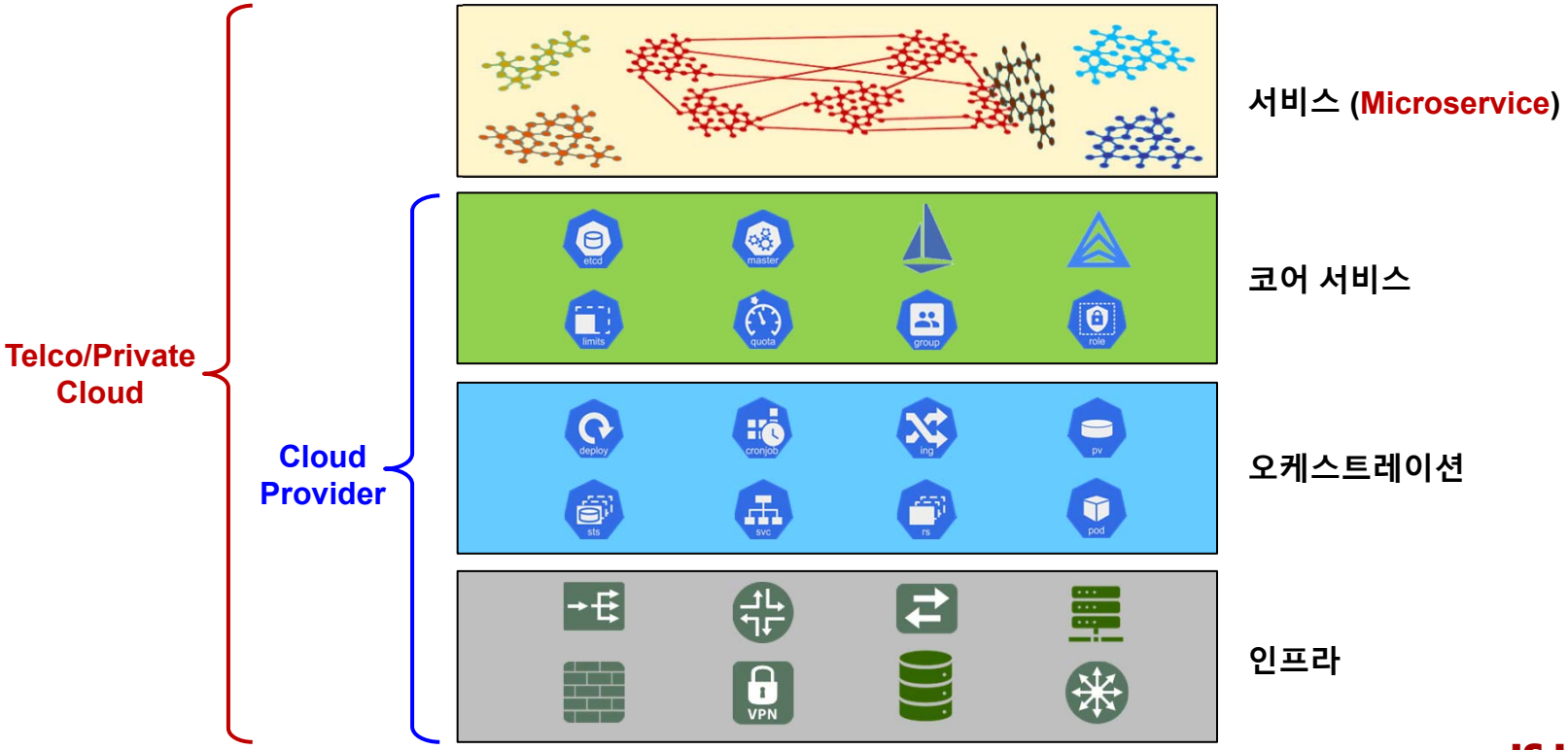
JS Lab

목차

james@jslab.kr

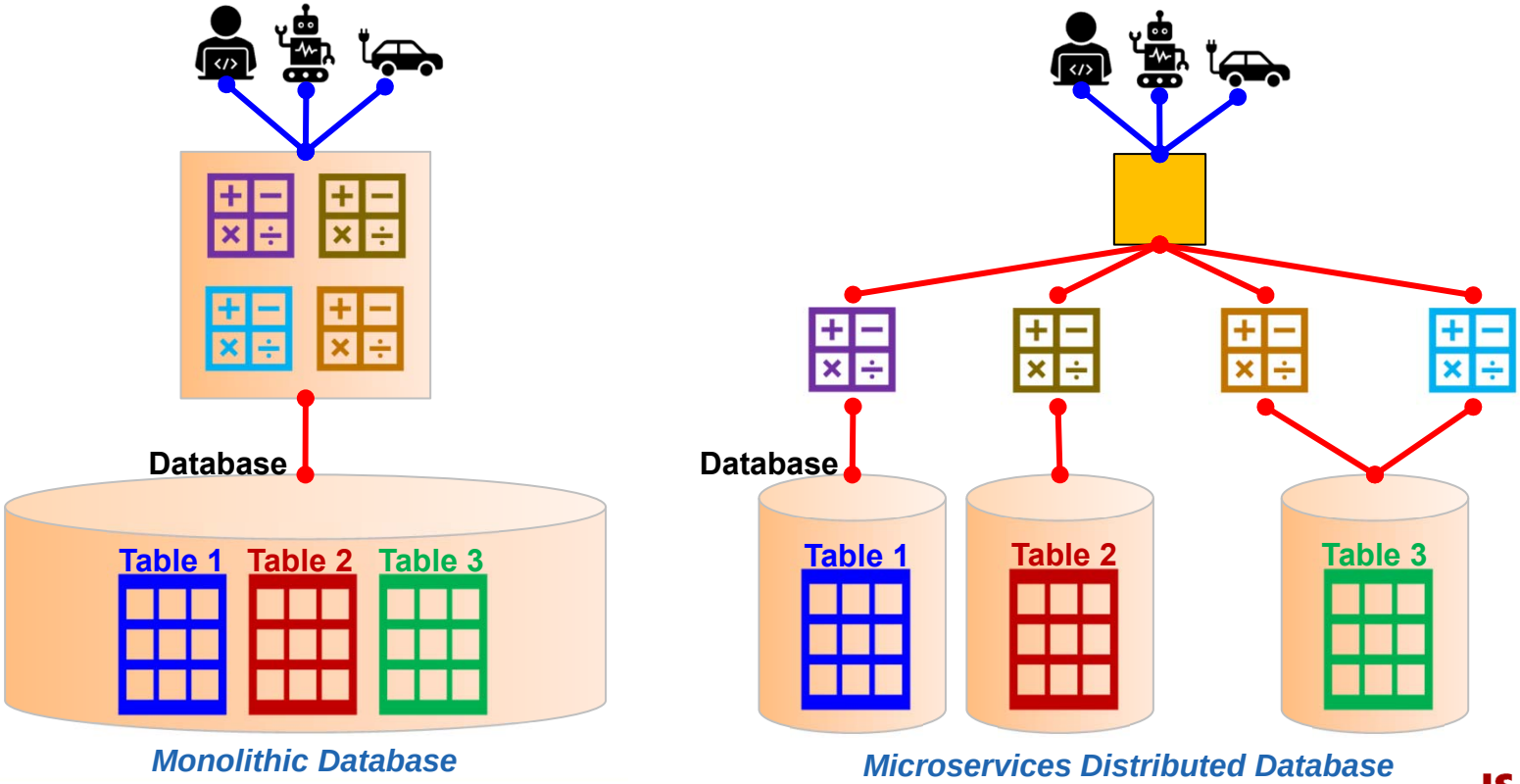
1장. 개요

- ‘Microservice’ @ Market
  - non-Cloud vs. Cloud
  - Public Cloud vs. Private Cloud
  - Service vs. Business



1장. 개요

- **from Monolithic to Microservices**
  - 분산 거버넌스/데이터 관리
  - 분산 비즈니스/메세징 규칙



james@jslab.kr

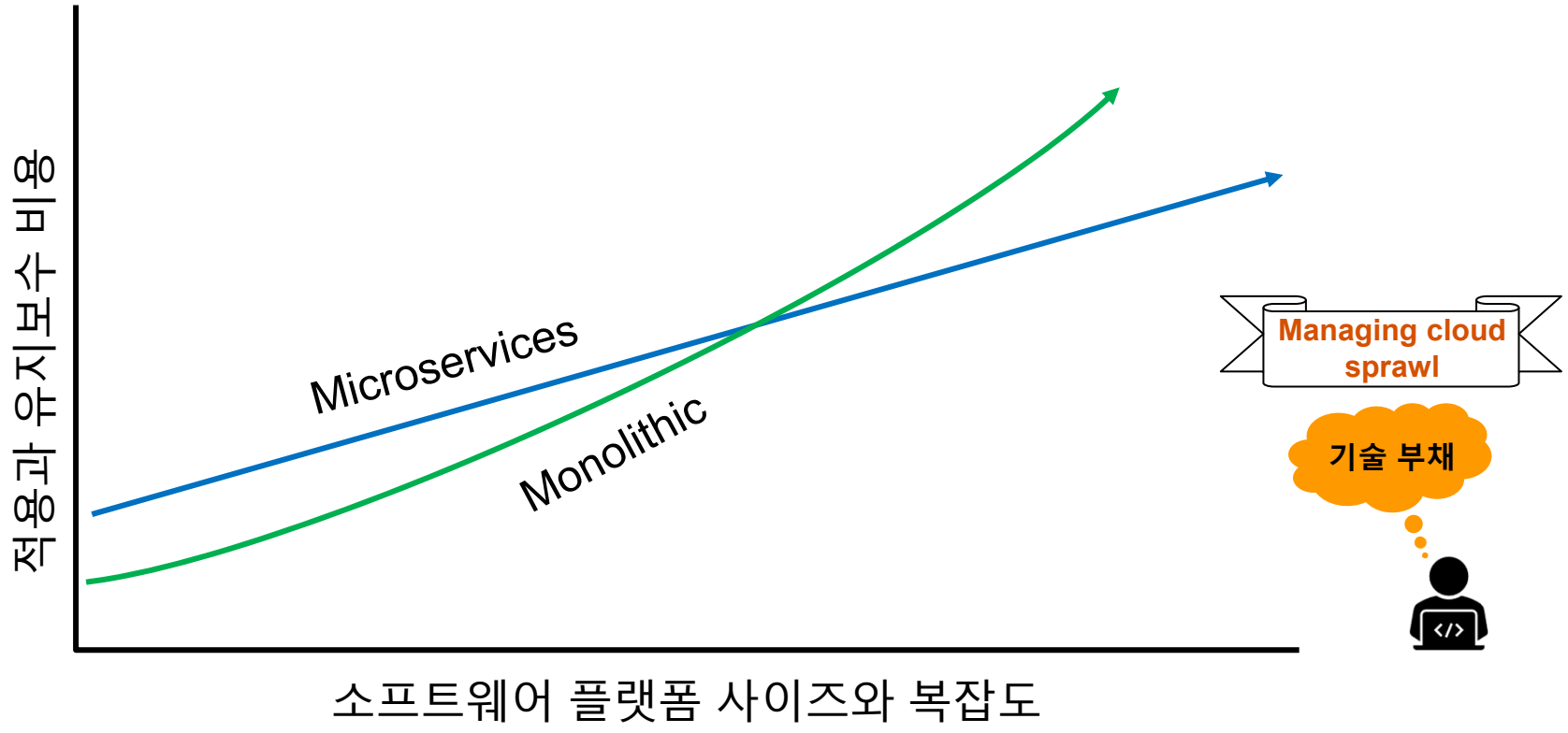
1장. 개요

- **Wikidipia: 마이크로서비스 아키텍처**
  - 마이크로서비스는 애플리케이션을 느슨히 결합된 서비스의 모임으로 구조화하는 서비스 지향 아키텍처(SOA) 스타일의 일종인 소프트웨어 개발 기법
  - MSA에서 서비스들과 프로토콜은 가벼운 편
  - 애플리케이션을 여러 서비스로 분해할 때의 장점은 모듈성을 개선, 애플리케이션의 이해, 개발, 테스트를 더 쉽게 해주고 애플리케이션 침식에 탄력적
  - 규모가 작은 자율적인 팀들이 팀별 서비스를 독립적으로 개발, 전개, 규모 확장을 할 수 있게 함으로써 병렬로 개발
  - 지속적인 리팩터링을 통해 개개의 서비스 아키텍처가 하나로 병합될 수 있게 허용
  - 마이크로서비스 기반 아키텍처는 지속적 배포와 전개(Deploy)

james@jslab.kr

1장. 개요

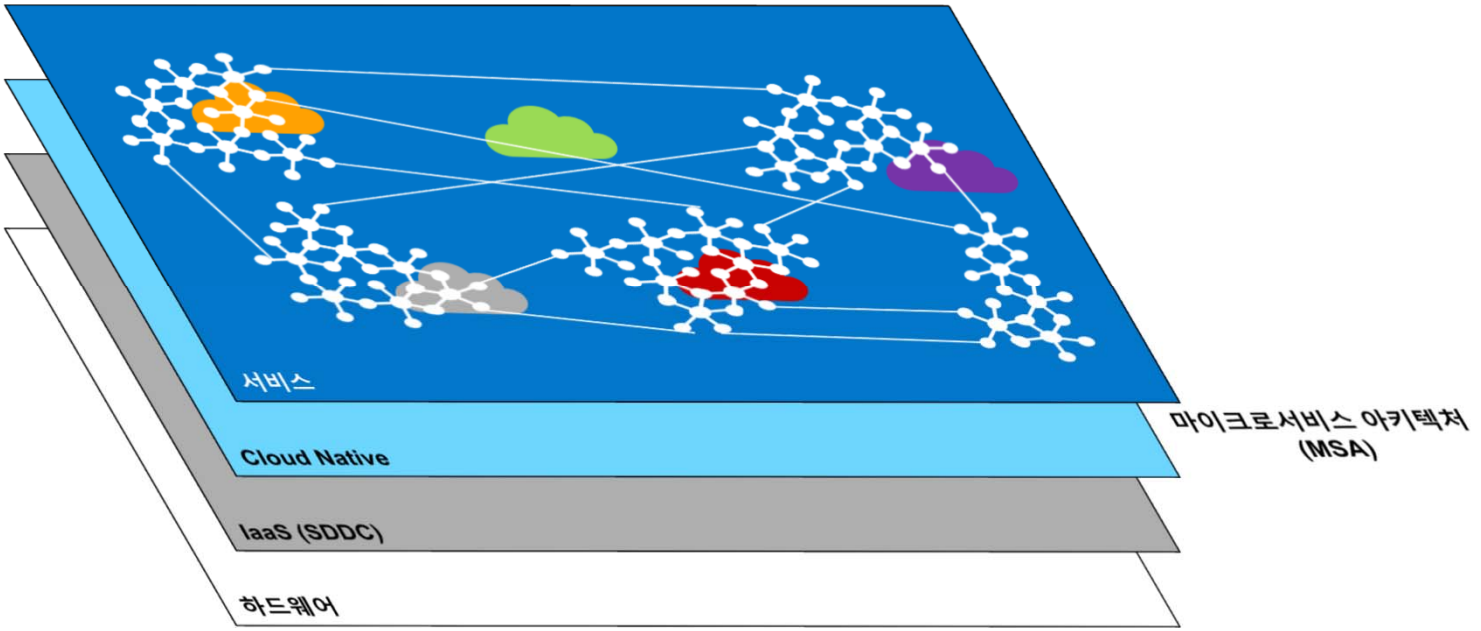
- 비용
- 규모 확장에 따른 장단점 고려



1장. 개요

계층화 인프라

- 인프라의 계층별 추상화 (서비스에 집중)
- 계층간 격리와 정책 기반 서비스 노출
- 성능 개선 (계층 Offload)
- 연결 호환성



SDDC (Software Defined Datacenter), **IaaS** (Infrastructure-as-a-Service), **MSA** (Microservice Architecture)

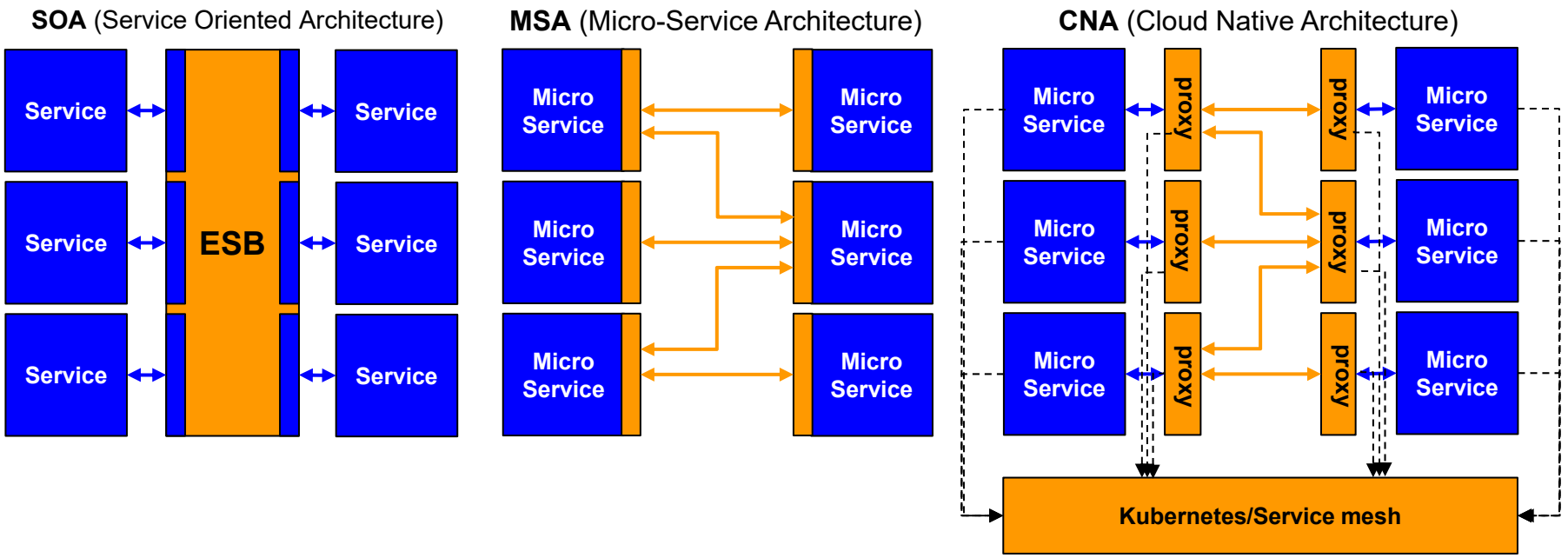
1장. 개요

- 하드웨어 계층
  - 협업과 융합 기술을 위한 고속의 클라우드 서비스용 아키텍처 구성
  - 기술 발전 수용과 클라우드 성능 향상을 지원하는 하드웨어 확장 구조
- IaaS 계층
  - 클라우드 서비스의 유연성을 제공하는 가상화 인프라 계층
  - 클라우드 서비스를 위한 Compliance 수용 보안 강화
- Cloud Native 계층
  - 생산성 증대를 위한 클라우드 서비스 제공
  - 멀티 클라우드 기반 확대 서비스
- 서비스 계층
  - 협동 연구 개발과 상용화를 위한 플랫폼 서비스 제공
  - 개발과 상용화 과정의 자동화 제공과 지식의 자산화

james@jslab.kr

1장. 개요

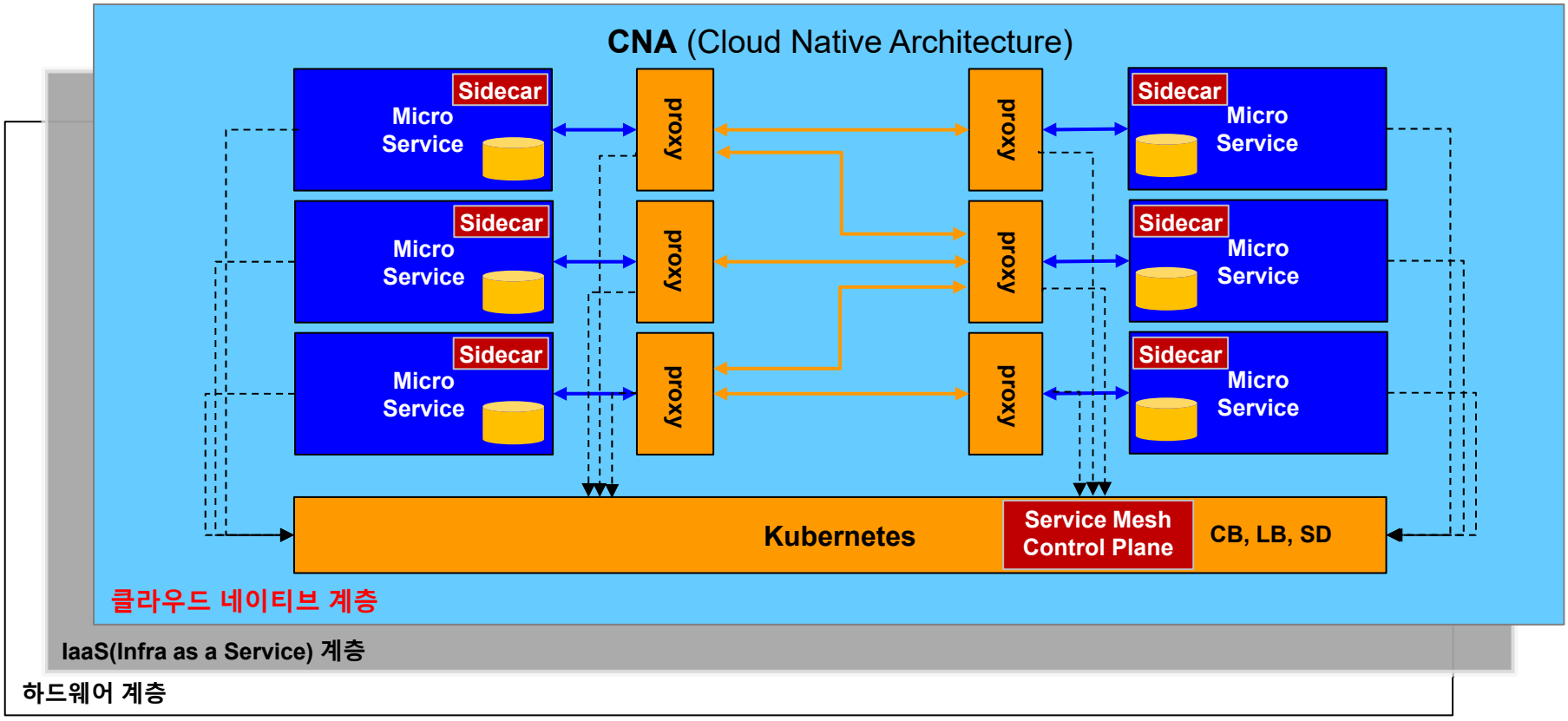
- Monolithic
- Service Oriented Architecture (SOA)
- Microservice architectures (MSA)
- Cloud Native Architecture (CNA)



1장. 개요

CNA 의 서비스 메쉬 관리

- Sidecar Design Pattern: 라우터 내장 CB, LB, SD 내장
- CNCF의 'Istio'는 정책 강화 Telemetry 제공

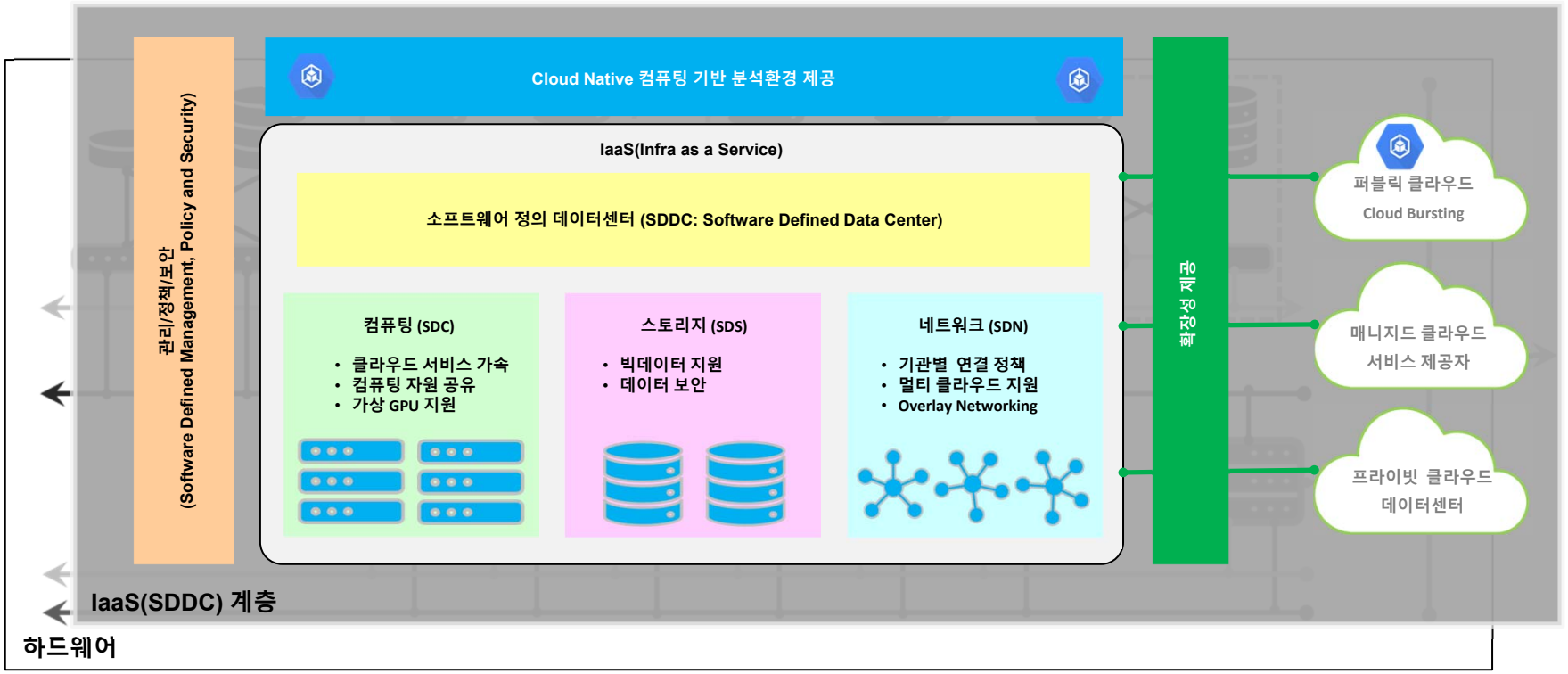


james@jslab.kr

CB (Circuit Breaker), **LB** (Load Balancer), **SD** (Service Discovery)

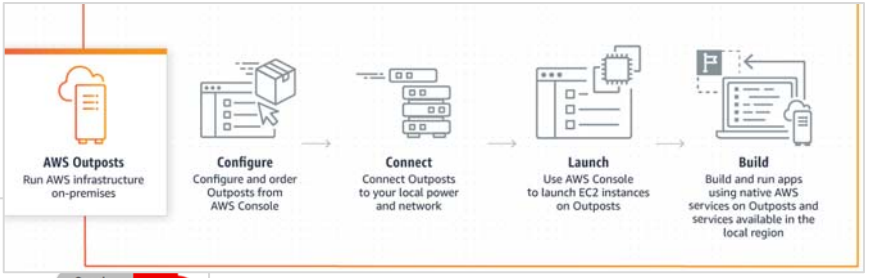
1장. 개요

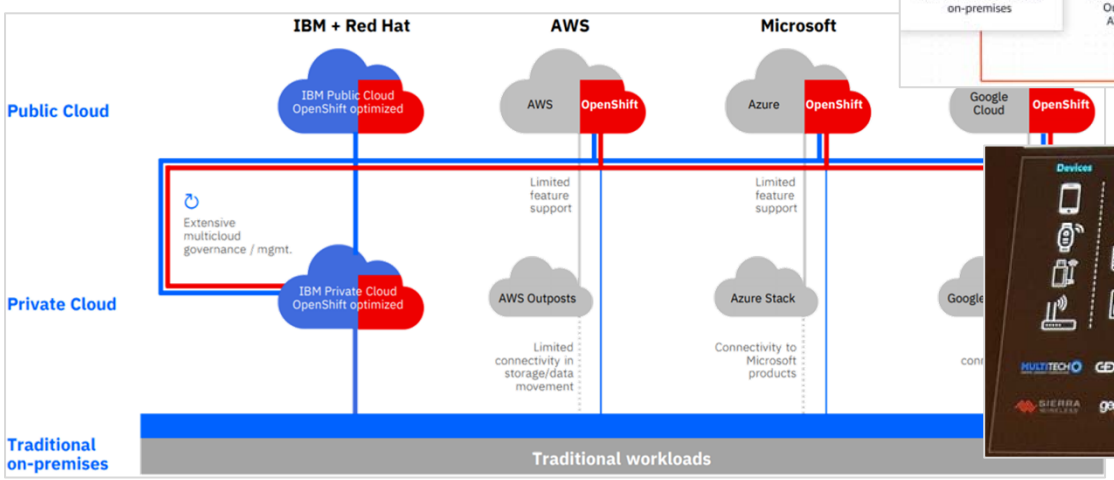
- 하드웨어의 추상화 기반 IaaS 인프라 운영 (SDDC)
- 가상화 자원 제공
- 하이브리드(Hybrid)/멀티(Multi) 클라우드 인프라 서비스

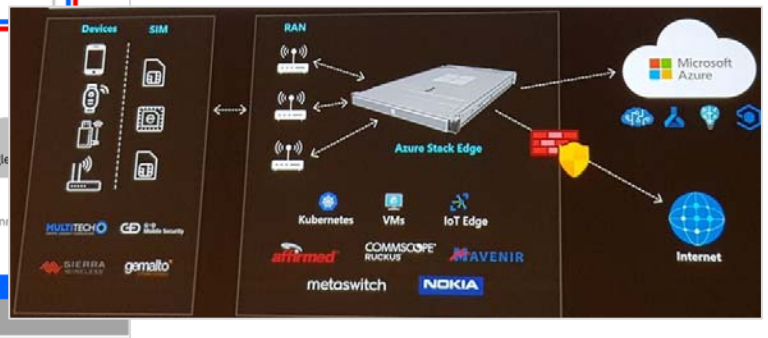


2장. 클라우드 서비스 (Cloud Services)

- 하이브리드/멀티 클라우드
  - **IBM Cloud Pak**
  - **AWS Outpost**
  - **MS Azure Stack** (Edge)
  - **Google Anthos**





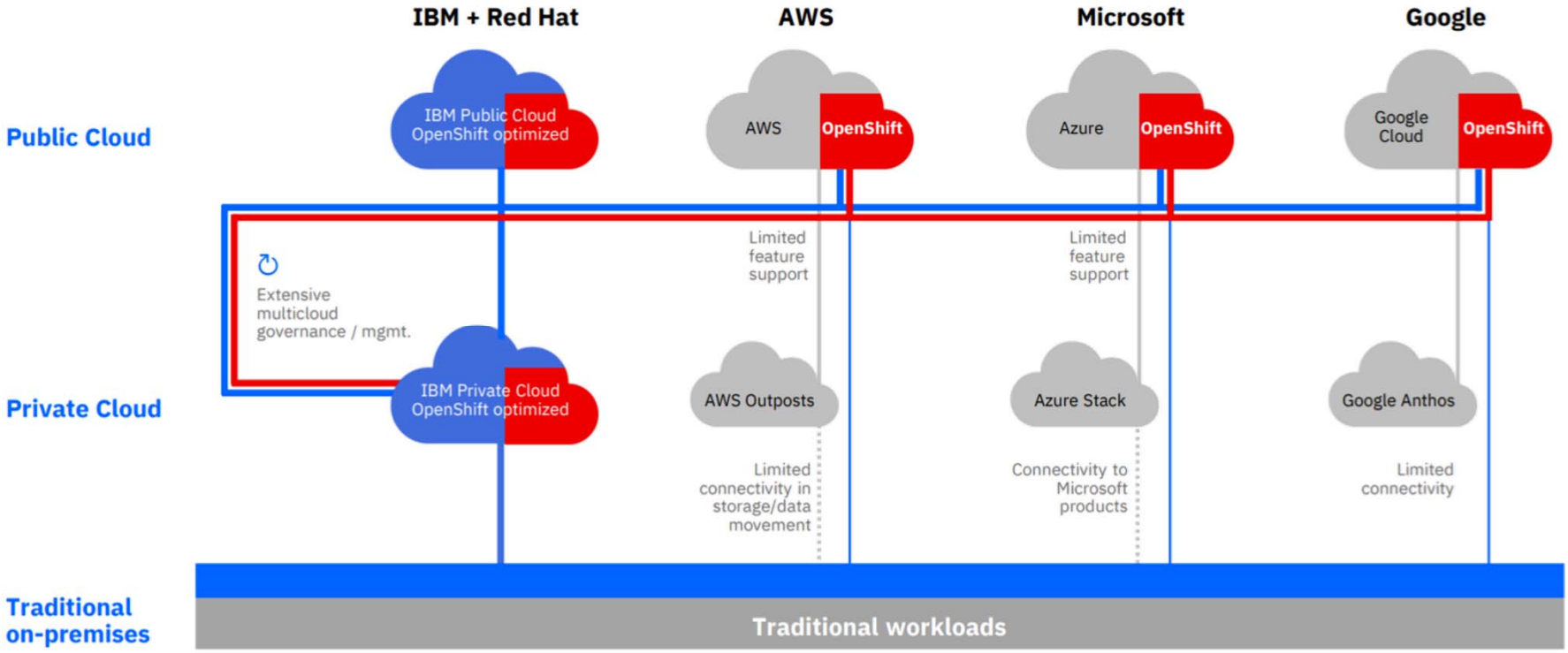


https://www.metricstream.com/technology/grc-cloud.htm

james@jslab.kr

2장. 클라우드 서비스 (Cloud Services)

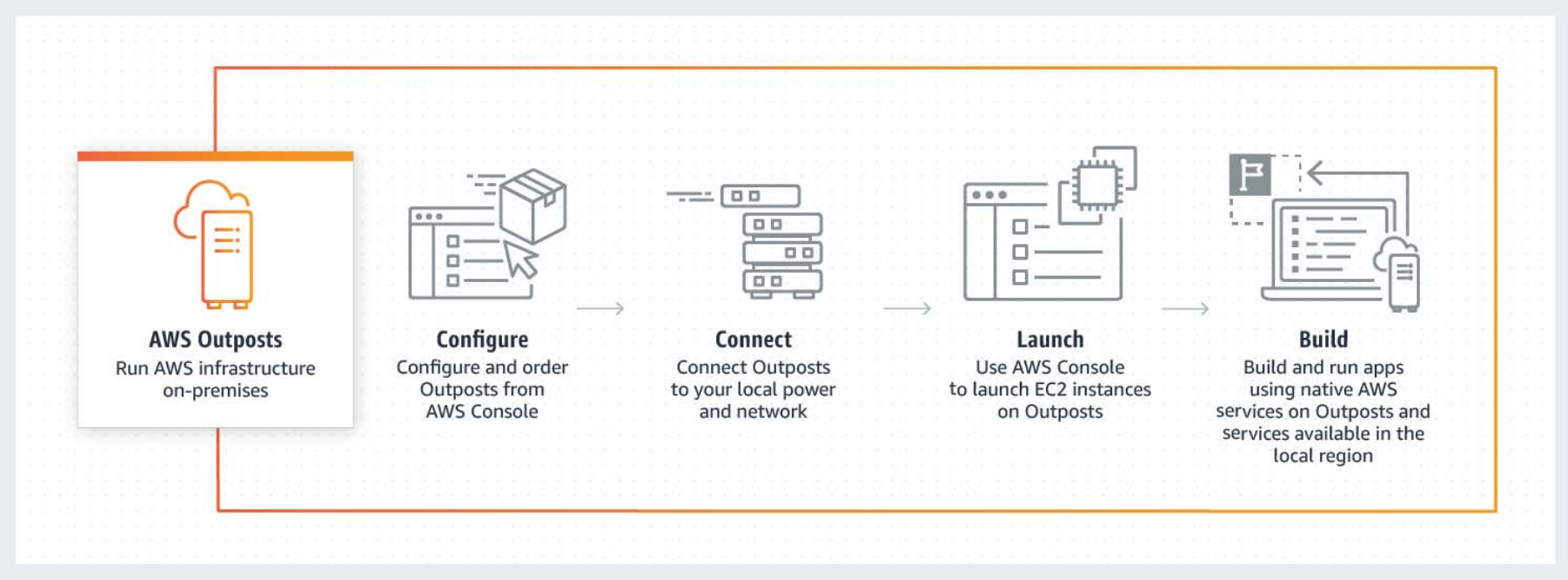
- **IBM Cloud Pak**
  - **Hybrid, Multicloud platform**
  - **Cloud Pak on OpenShift**
  - **Cloud native for all clouds - middleware anywhere**



https://www.metricstream.com/technology/grc-cloud.htm

2장. 클라우드 서비스 (Cloud Services)

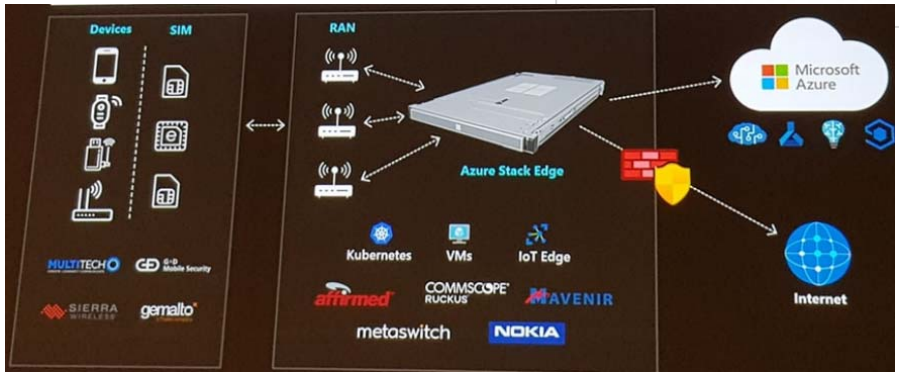
- **AWS Outpost**
  - **AWS 온프레미스 확장**
  - **관리 플레인 선택: AWS 네이티브 변형 or VMware Cloud on AWS**
  - **완전관리형**



https://aws.amazon.com/ko/outposts/

2장. 클라우드 서비스 (Cloud Services)

- **MS Azure Stack**
  - **Azure Stack Edge: ML, RAN, IoT**
  - **Azure Stack HCI**
  - **Azure Stack Hub**

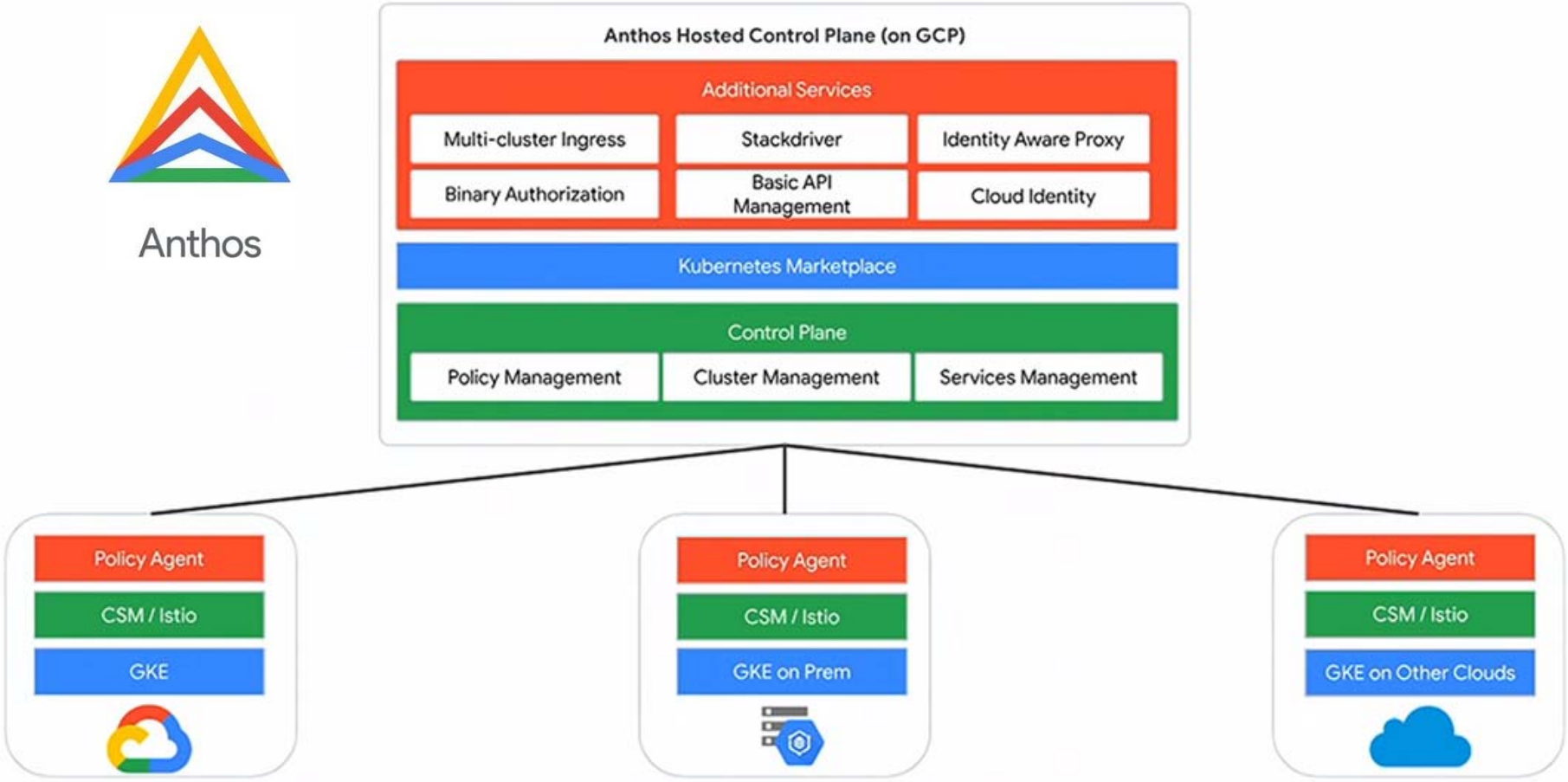


https://azure.microsoft.com/ko-kr/overview/azure-stack/

james@jslab.kr

2장. 클라우드 서비스 (Cloud Services)

- **Google Anthos**

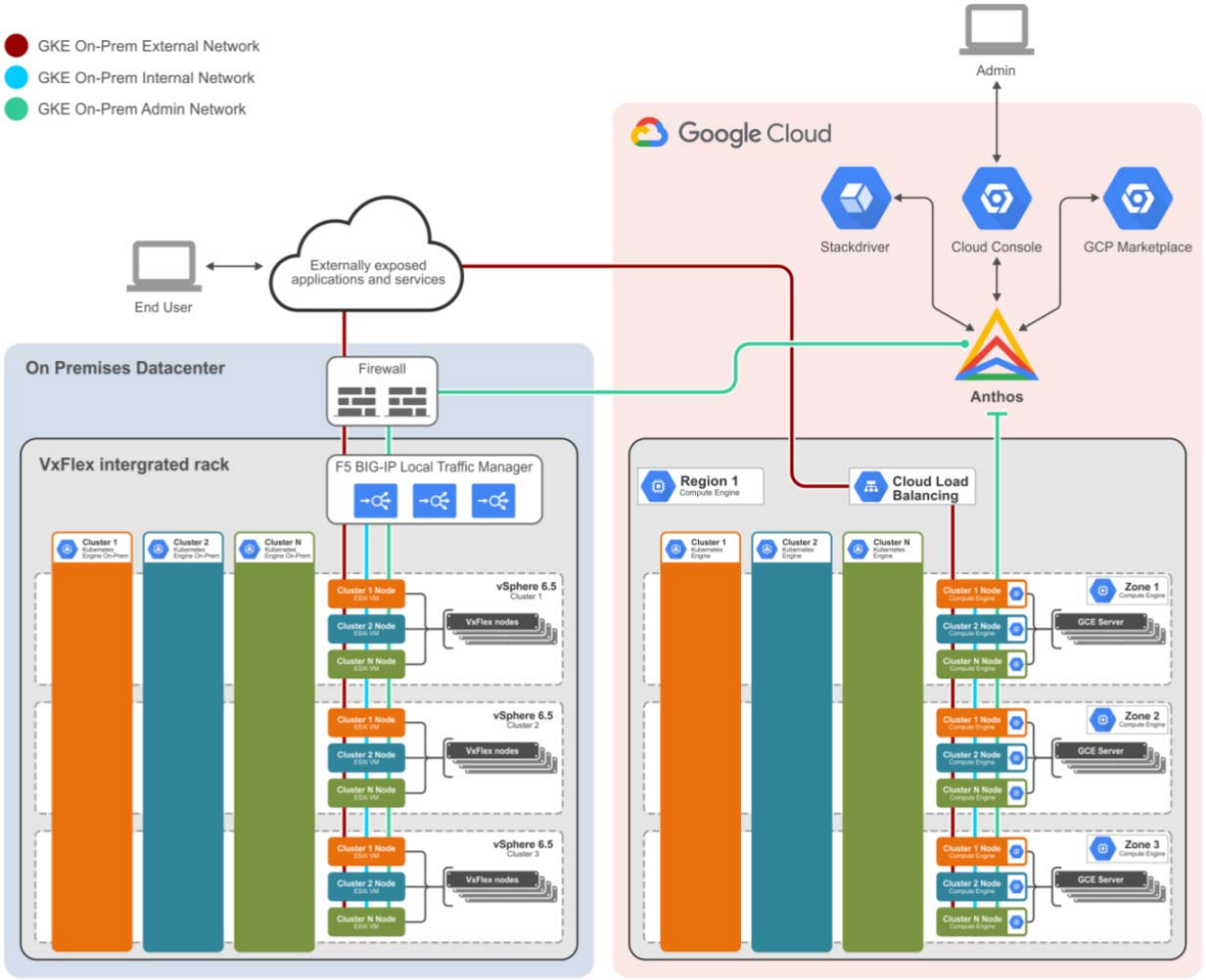


https://www.bespinglobal.com/google-anthos-multi-cloud/

https://blog.aquasec.com/hybrid-cloud-security-google-anthos

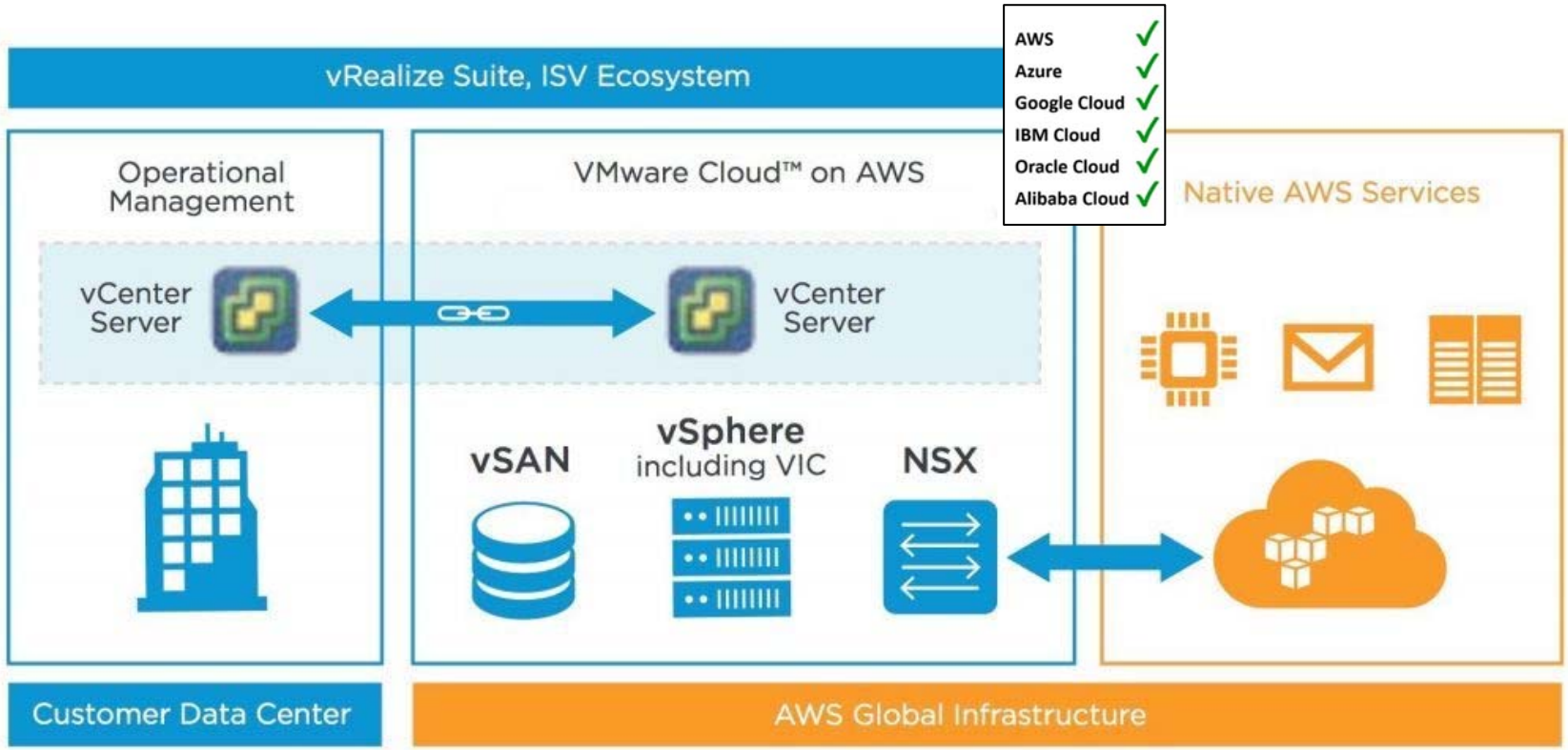
2장. 클라우드 서비스 (Cloud Services)

- **VxFlex integrated rack for Google Cloud's Anthos**



2장. 클라우드 서비스 (Cloud Services)

- **VMware Cloud**
  - **VMware Cloud on AWS**
  - **지원 확장: Azure, Google, IBM, Oracle, Alibaba and KT Cloud**



https://www.actualtech.io/getting-hands-dirty-installing-vmware-cloud-aws/

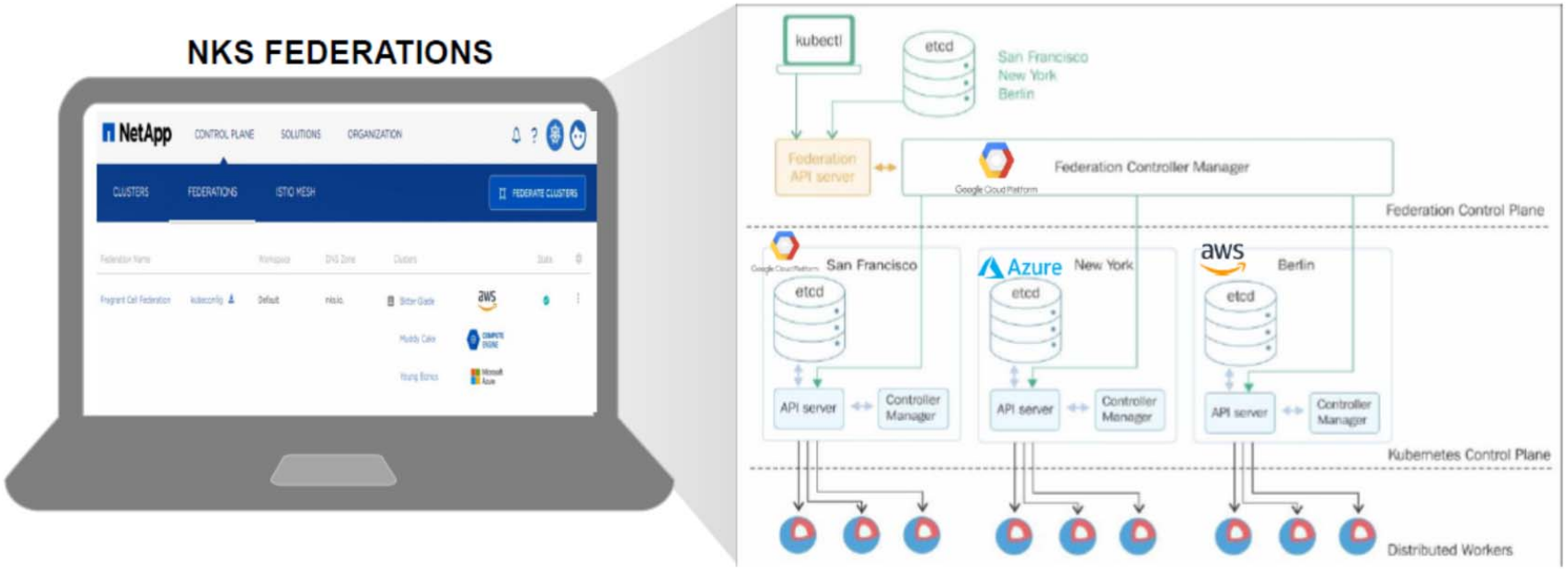
2장. 클라우드 서비스 (Cloud Services)

- **Rancher Labs**
  - **Rancher 2.x**

https://rancher.com/announcing-rancher-2-0/

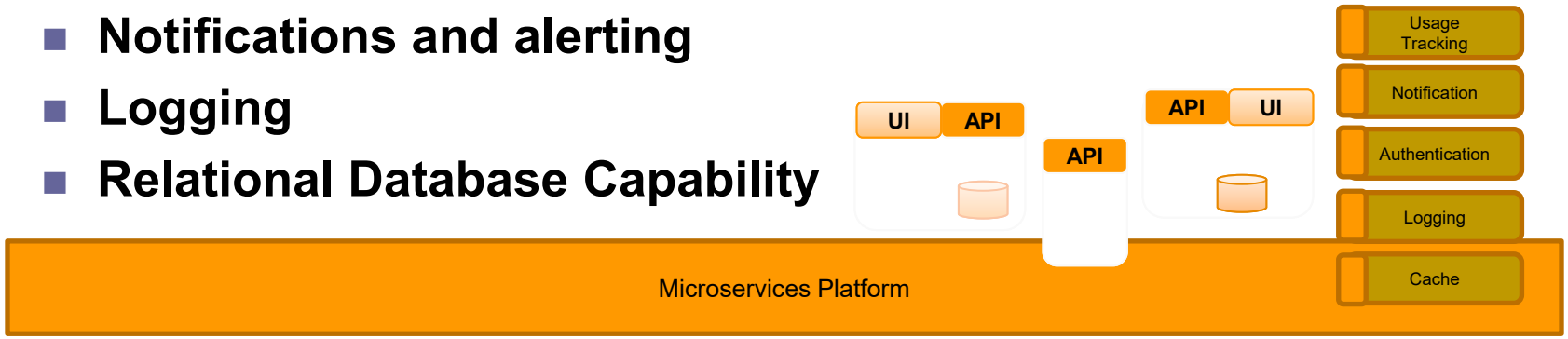
2장. 클라우드 서비스 (Cloud Services)

- **NKS (NetApp Kubernetes Service)**
  - **Multi Kubernetes Cluster Multi Federation**



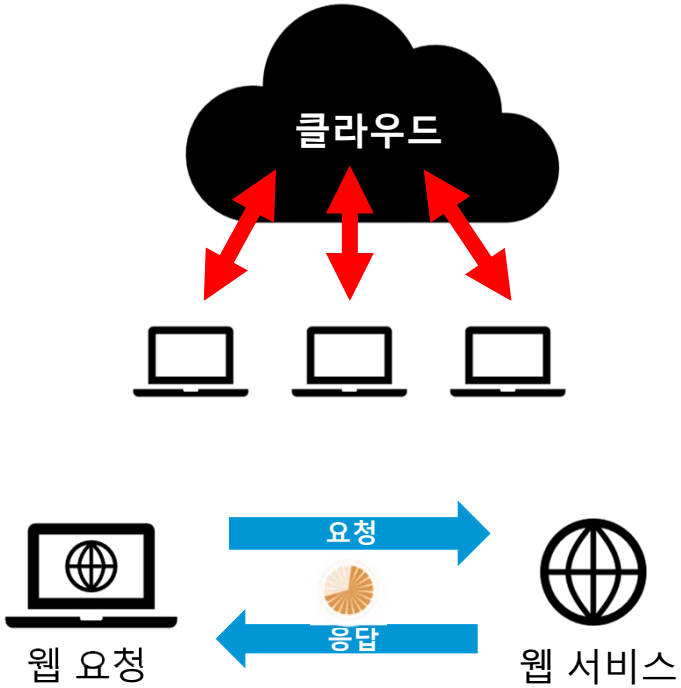
2장. 클라우드 서비스 (Cloud Services)

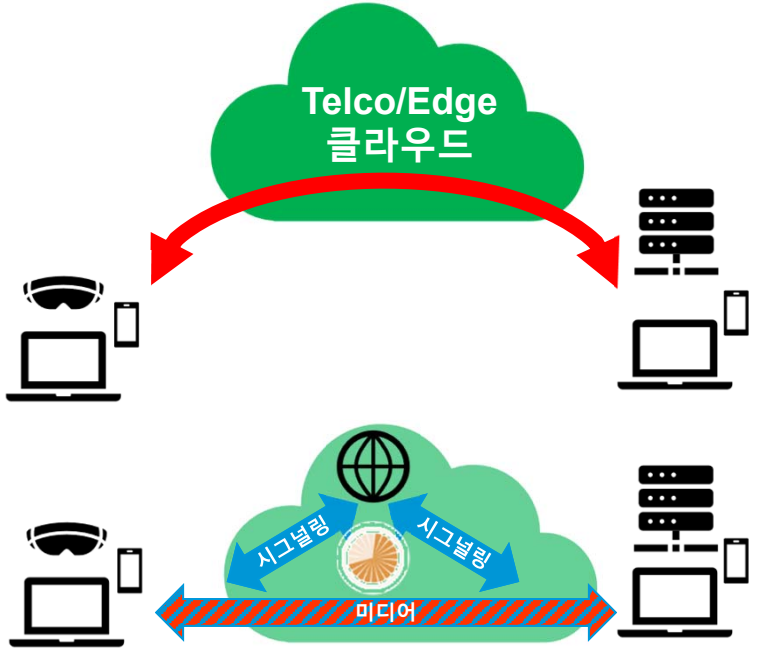
- **Microservices cross platform Capabilities**
  - **Authentication**
  - **Persistent Storage**
  - **Cache**
  - **Load balancing/API Gateway**
  - **Discovery/Lookup**
  - **Monitoring**
    - ✓ Functional/Business KPIs
    - ✓ Non Functional Platform/Container & Infra
  - **Audit, Usage tracking, Billing**
  - **Notifications and alerting**
  - **Logging**
  - **Relational Database Capability**



2장. 클라우드 서비스 (Cloud Services)

- **Public vs. Telco 클라우드**
  - **Public Cloud: 소프트웨어 정의 가상 인프라 기반 서비스 (웹서비스)**
  - **Telco/Edge Cloud: 언더레이 인프라 기반 클라우드 서비스**
  - **Telco/Edge Cloud는 하드웨어 인프라 환경 고려**



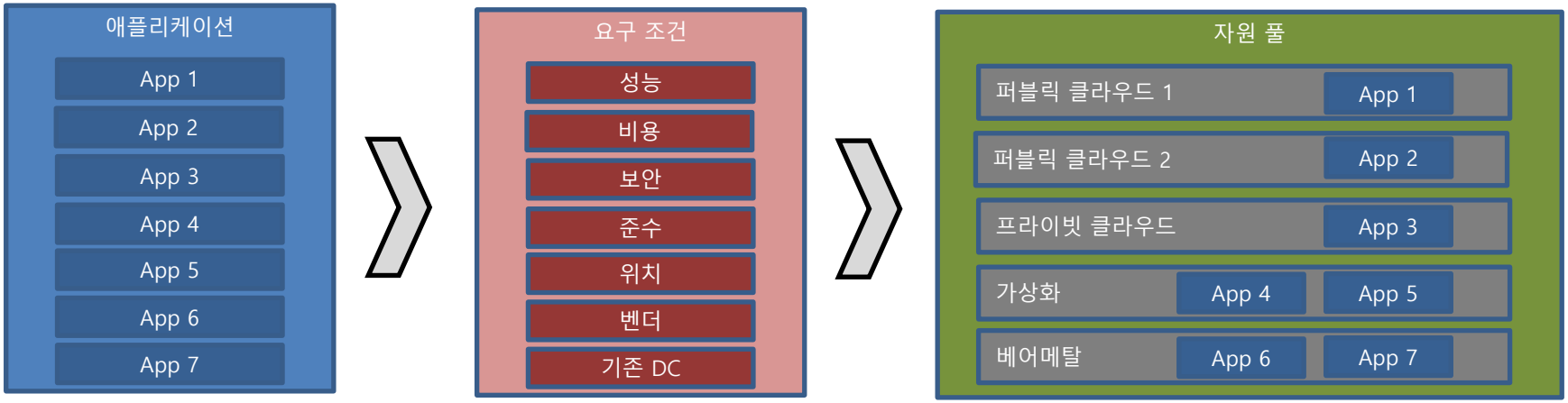


실시간 멀티미디어 애플리케이션
(고화질 스트리밍, 고화질 게임, AR, VR 등을 위한 경로)

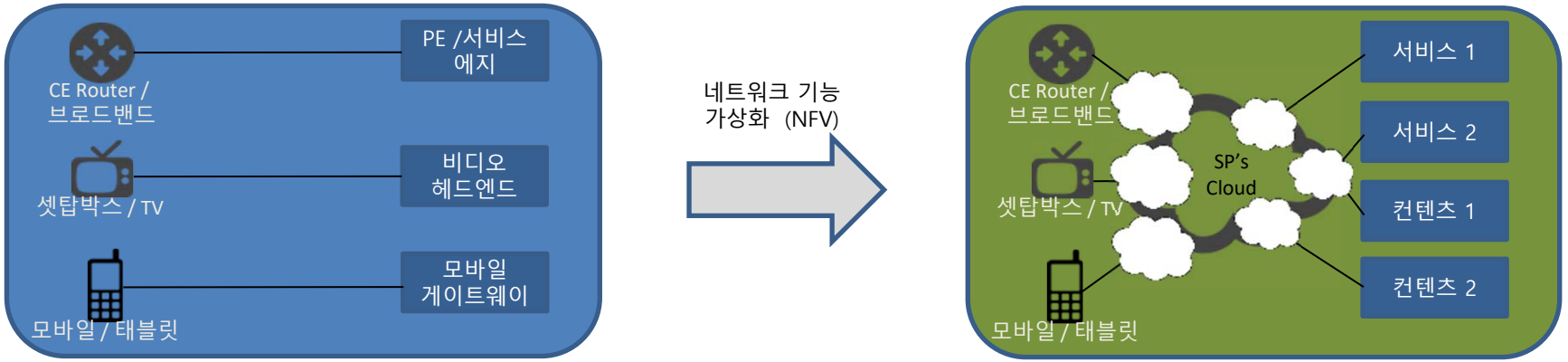
james@jslab.kr

2장. 클라우드 서비스 (Cloud Services)

- 엔터프라이즈는 애플리케이션 요구에 적합한 클라우드 자원

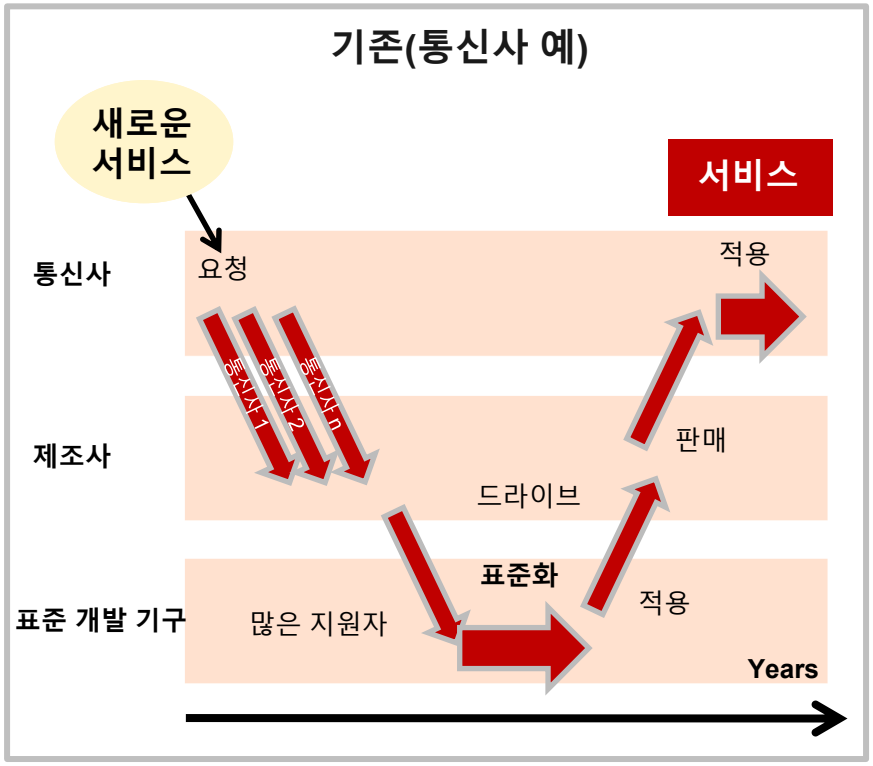


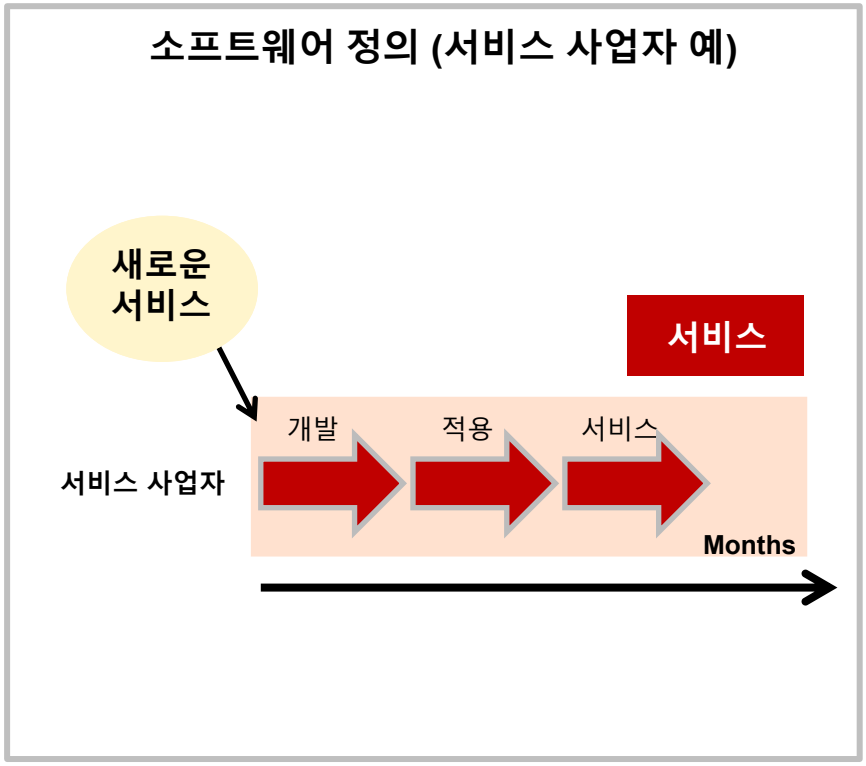
- 텔코 (Telco)는 네트워크 기능 가상화 기반 데이터센터화



2장. 클라우드 서비스 (Cloud Services)

- 통신사와 서비스 사업자의 적용 환경
- 개발능력 미보유 엔터프라이즈는 서비스 제공 가능 오픈소스나 제조사의 SDx 솔루션 필요





2장. 클라우드 서비스 (Cloud Services)

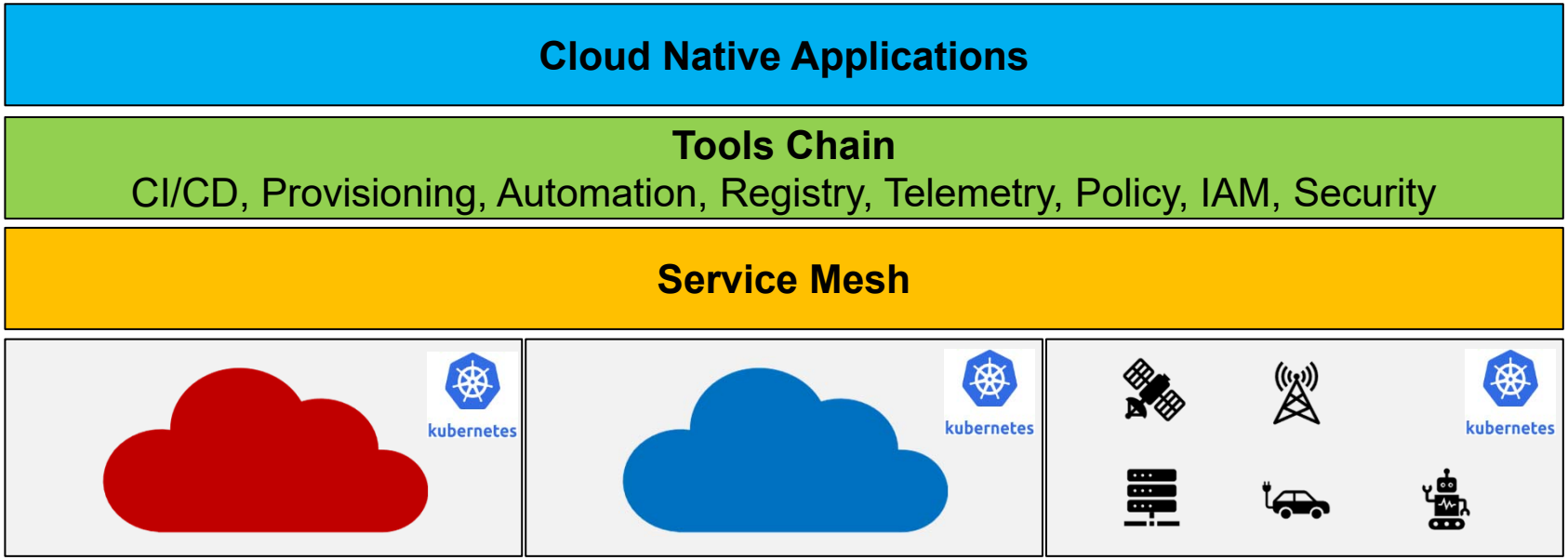
Edge vs Core @ Telco

	Edge (에지) 클라우드	중앙 클라우드
App의 위치	노드의 물리적 위치에서 중요한 서비스	비교적 위치와 독립적인 서비스
워크로드의 이동성	워크로드가 노드간 이동	클라우드 노드 장애 이외에는 비교적 고정
워크로드의 역동성	다양한 App들이 다양한 시간에 크게 다른 요구를 함	서비스를 적용하면 대부분의 시간에 안정적인 워크로드
아키텍쳐	다른 형태의 많은 수의 노드와 다양한 용량과 기술	대부분 동일하며 차이가 작음 (예: AWS, OpenStack, Azure 등)
지연	지연과 거리는 종단 사용자들을 위한 주요 역할	대부분 지연에 민감하지 않음
자원 가용성	에지노드는 작고, App을 위한 자원의 가용성을 보장하지 않음	가용성 확보는 중요하며 주요 기능 중 1개

james@jslab.kr

2장. 클라우드 서비스 (Cloud Services)

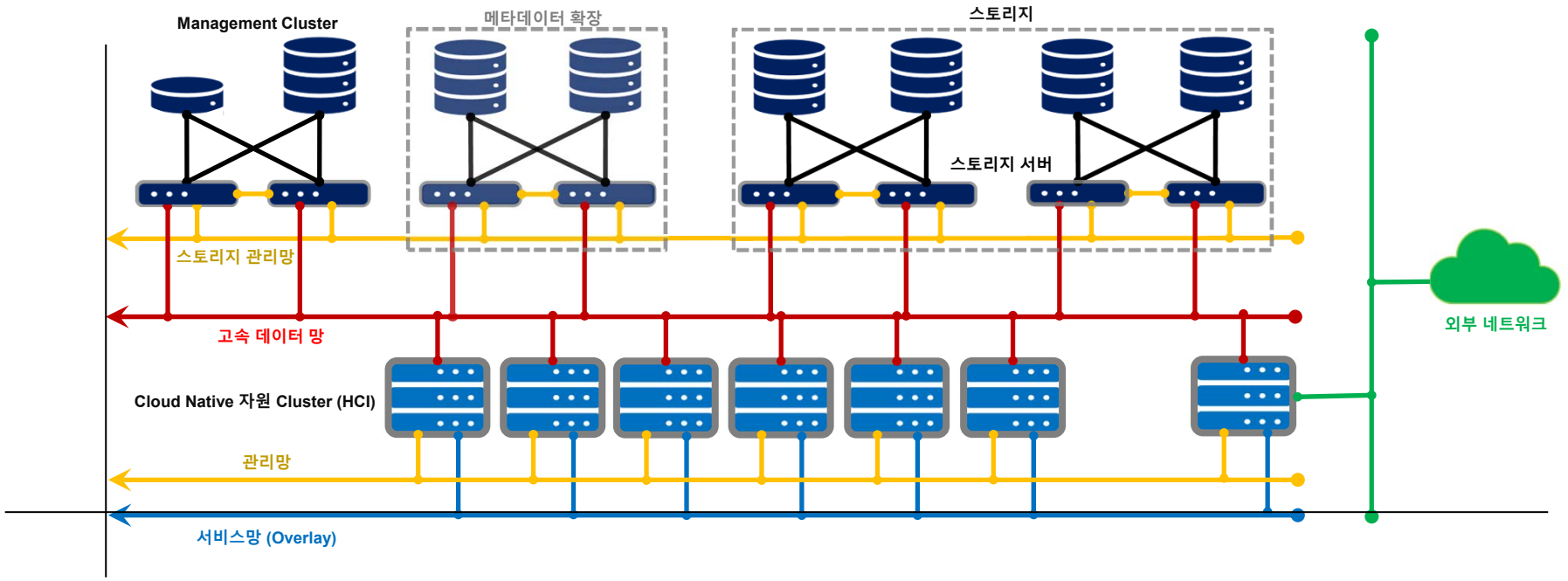
- **멀티 클라우드 Federation w/K8s**
  - **High Availability**
  - **Application Migration**
  - **Policy Enforcement**
  - **Vendor Lock-in Avoidance**
  - **Capacity Overflow**



james@jslab.kr

3장. MSA @ Private Cloud

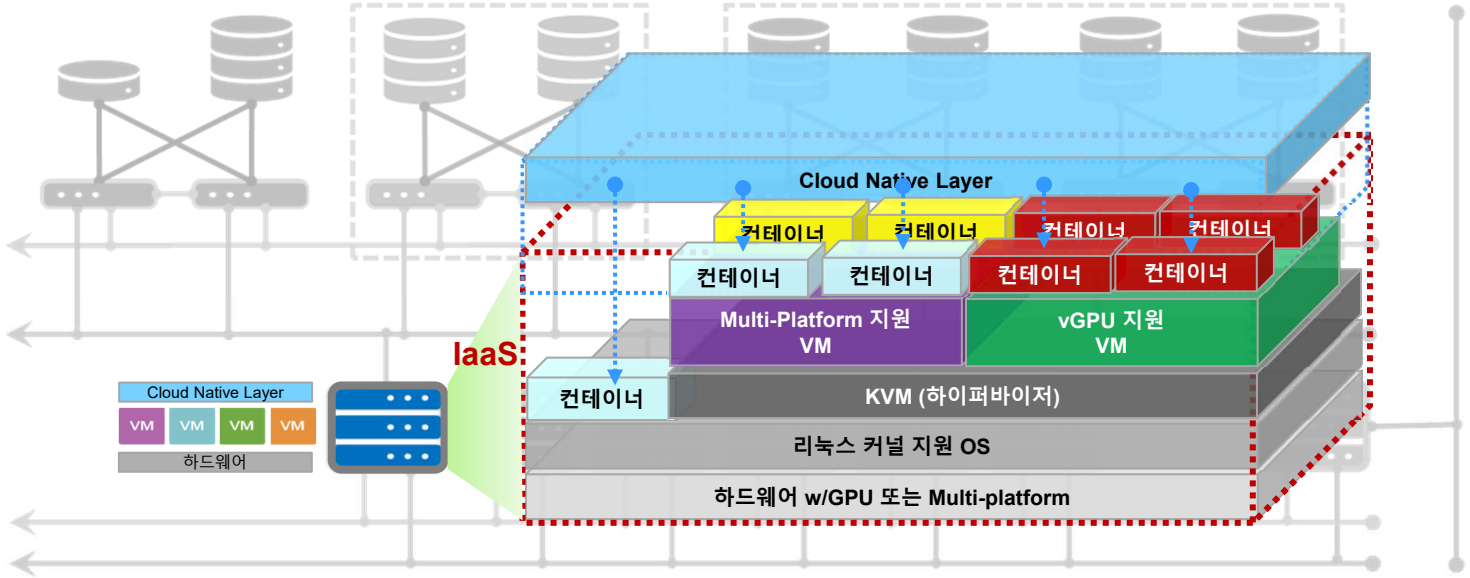
- 고속 처리 이중화 기반 스토리지와 HCI 기반 클러스터링
- IaaS(Infrastructure as a Service)를 위한 가상화 OS
- 클라우드 서비스 성능을 위한 스케일아웃(Scale-out) 확장



james@jslab.kr

3장. MSA @ Private Cloud

- IaaS(Infrastructure as a Service) 계층 (1 of 2)
- 컴퓨팅 노드 가상화
  - NUMA(Non-Uniform Memory Access) Aware
  - vGPU 지원 VM 지원 제공
  - Container화 OpenStack의 Kubernetes 기반 제어

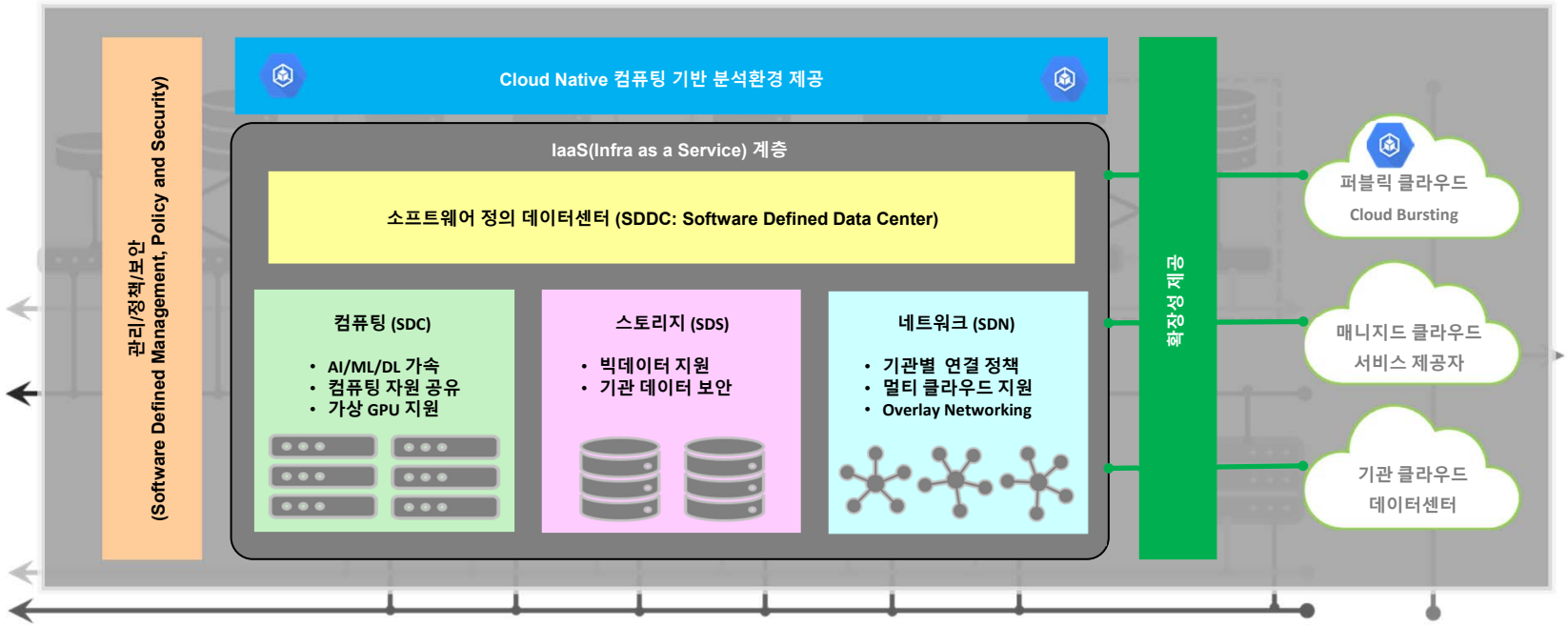


james@jslab.kr

3장. MSA @ Private Cloud

IaaS(Infrastructure as a Service) 계층 (2 of 2)

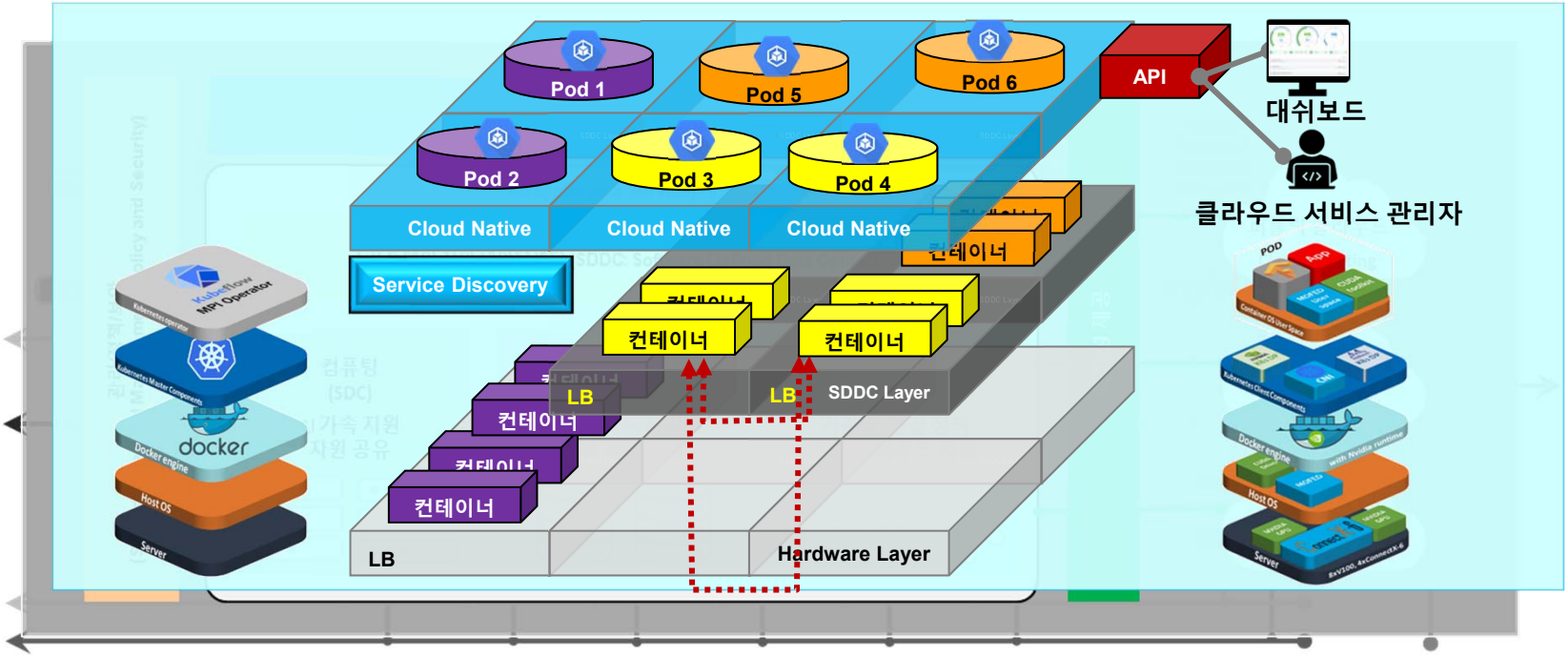
- 하드웨어 추상화 자원 제공으로 유연한 인프라 운영 확대
- SDDC 기반 컨테이너 보안 강화 클라우드 인프라 서비스
- Cloud Native 지원



3장. MSA @ Private Cloud

Cloud Native 계층 (1 of 2)

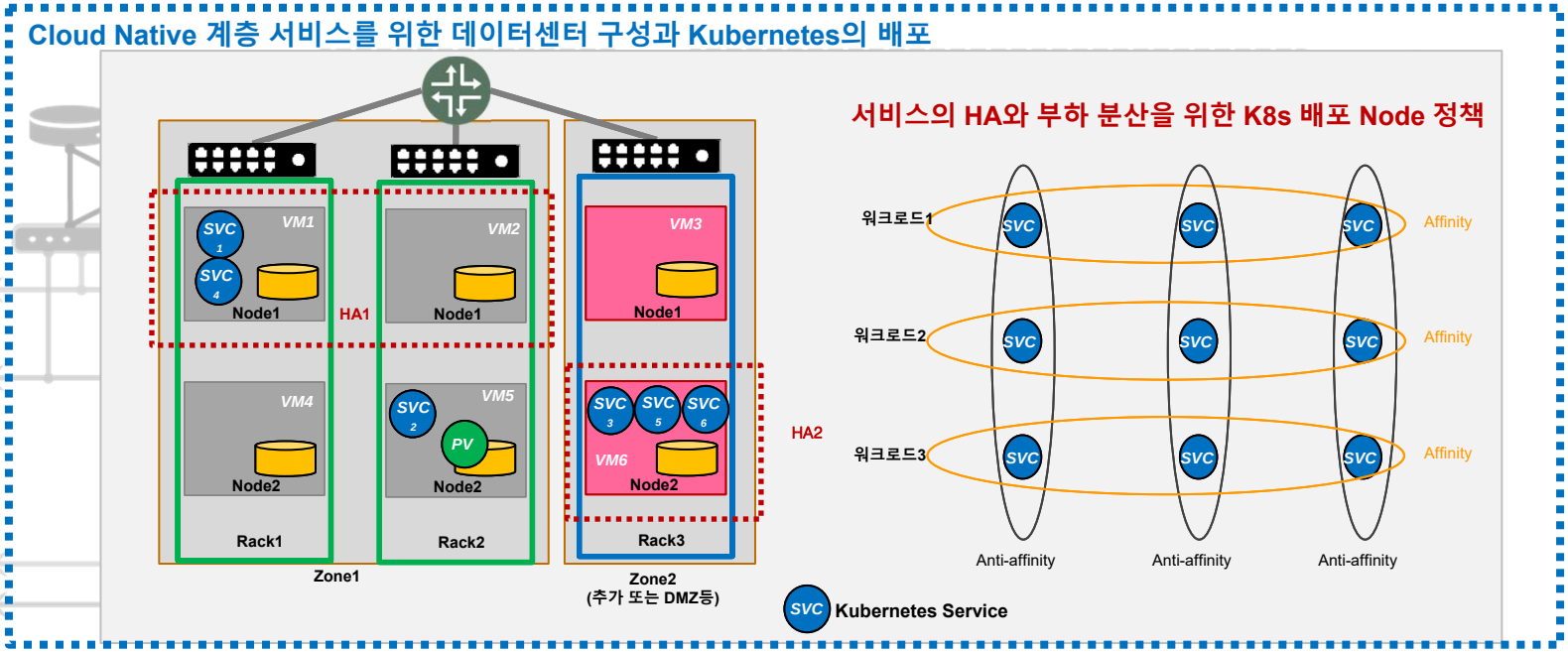
- Kubernetes는 자원 배포 단위로 Pod를 사용하여 추상화
- Pod는 Container로 구성, 제조사는 추상화 지원



3장. MSA @ Private Cloud

Cloud Native 계층 (2 of 2)

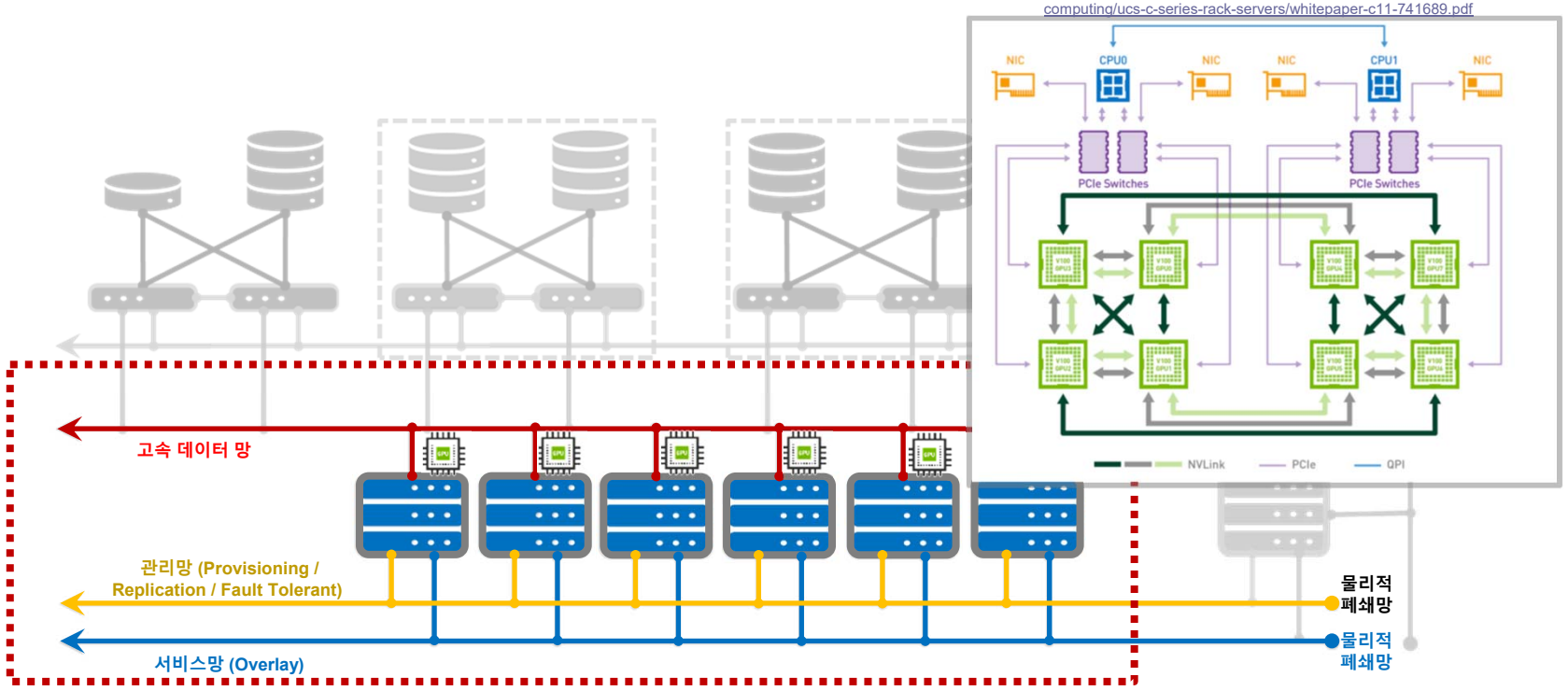
- HA용 동일 하드웨어/OS/컨테이너 사용의 Rack 구성
- K8s 추상화 PV(Persistent Volume)에 스토리지 매핑
- vGPU, vCPU 제공 VM 사용, 또는 BareMetal 사용 배포



GPU Clustering 구성

- 컴퓨팅 노드 내의 GPU 간 단축 경로 제공(CPU Bypass)
- 클러스터 내의 컴퓨팅 노드간 단축 경로 제공(Offload)
- 노드내 구성은 하드웨어 제조사 선택에 따라 다를 수 있음

https://www.cisco.com/c/dam/en/us/products/collateral/servers-unified-computing/ucs-c-series-rack-servers/whitepaper-c11-741689.pdf

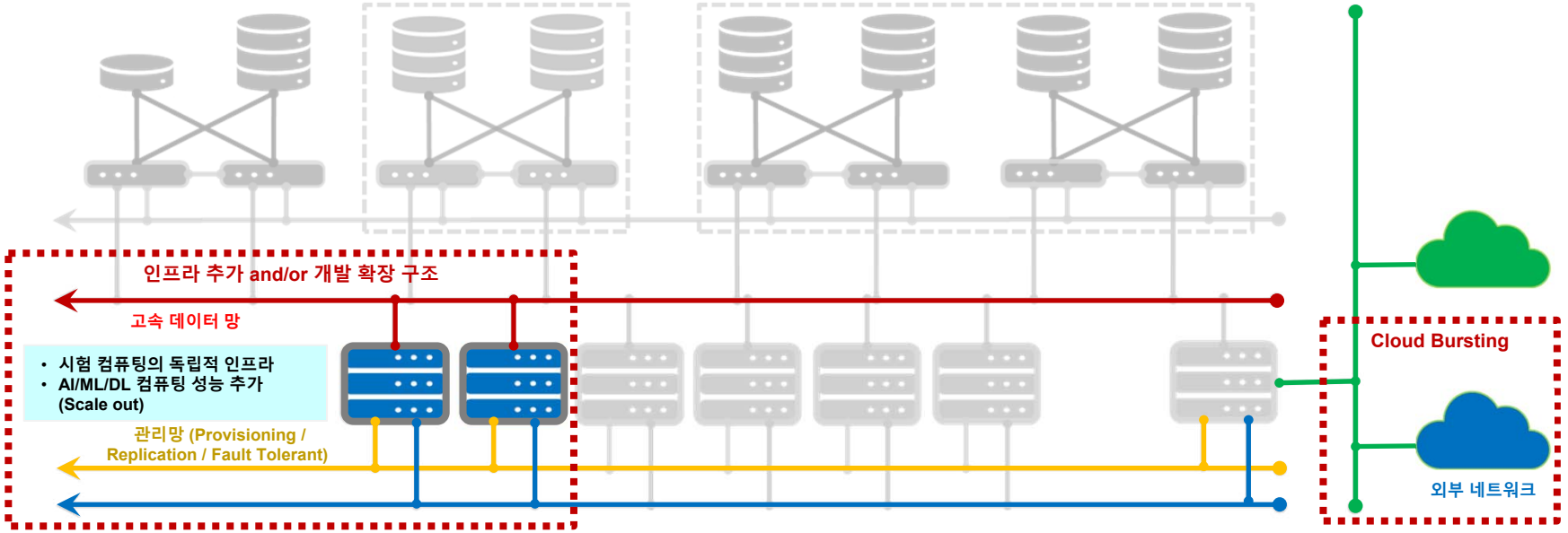


james@jslab.kr

3장. MSA @ Private Cloud

성능 확장 인프라

- Cloud Native Architecture(CNA) 기반 Scale out 확장
- 운영과 독립적인 ML/DL/AI 플랫폼 연구/개발 확장 환경
- Cloud bursting 확장

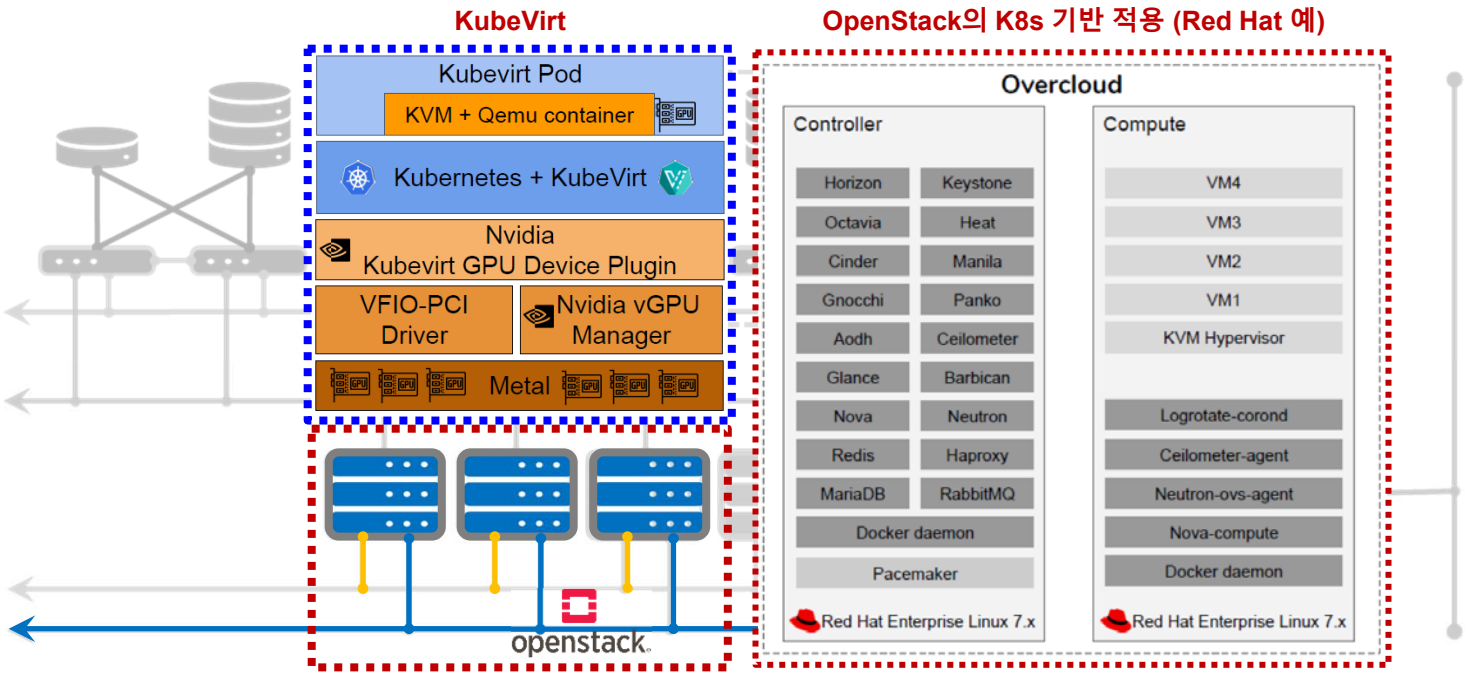


james@jslab.kr

3장. MSA @ Private Cloud

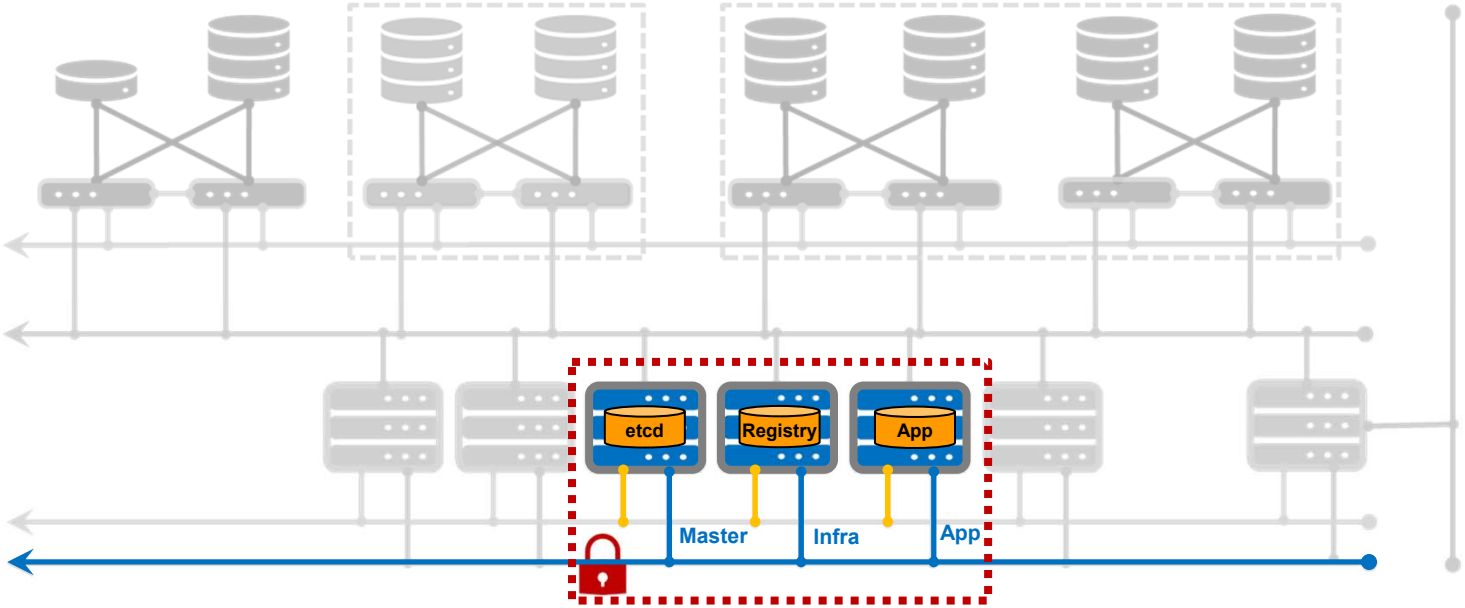
성능 확장 인프라

- Kubernetes Clustering VM Node 배포
- OpenStack 또는 KubeVirt 사용 가능
- 제어기능 들의 Packaging 설치를 위한 Helm 사용



3장. MSA @ Private Cloud

- 컨테이너 기반 보안 강화 플랫폼 구성
  - Cloud Native Architecture(CNA)를 위한 환경 플랫폼
  - 주요 요소 3중화 노드의 보안 관리와 네트워크의 노출 정책
  - ‘etcd’용 Master Node (x3), ‘Registry’용 Infrastructure Node (x3), ‘App’용 Application Node (x3)

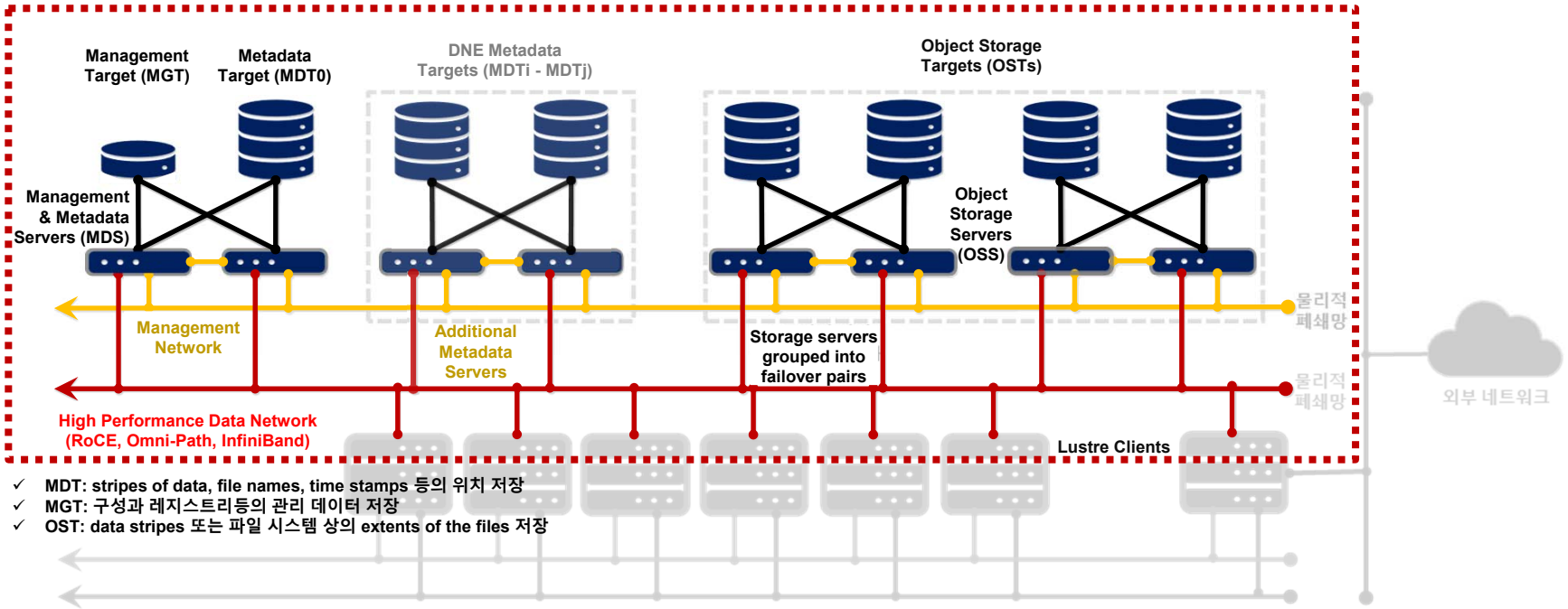


james@jslab.kr

3장. MSA @ Private Cloud

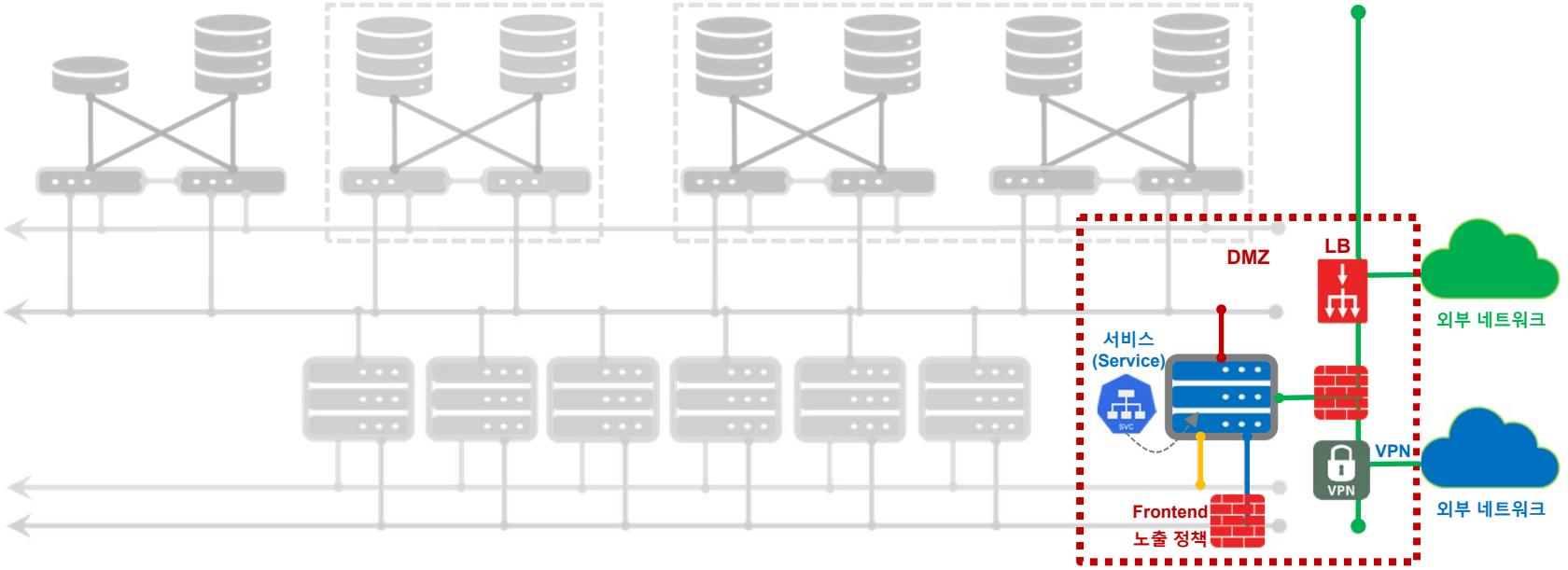
Lustre Storage 구성

- 스토리지 영역을 위한 초고속 빅데이터 처리 구성
- Clients: 파일 위치 결정을 위한 MDS 접속과 데이터 R/W를 위해 OSS에 접속



3장. MSA @ Private Cloud

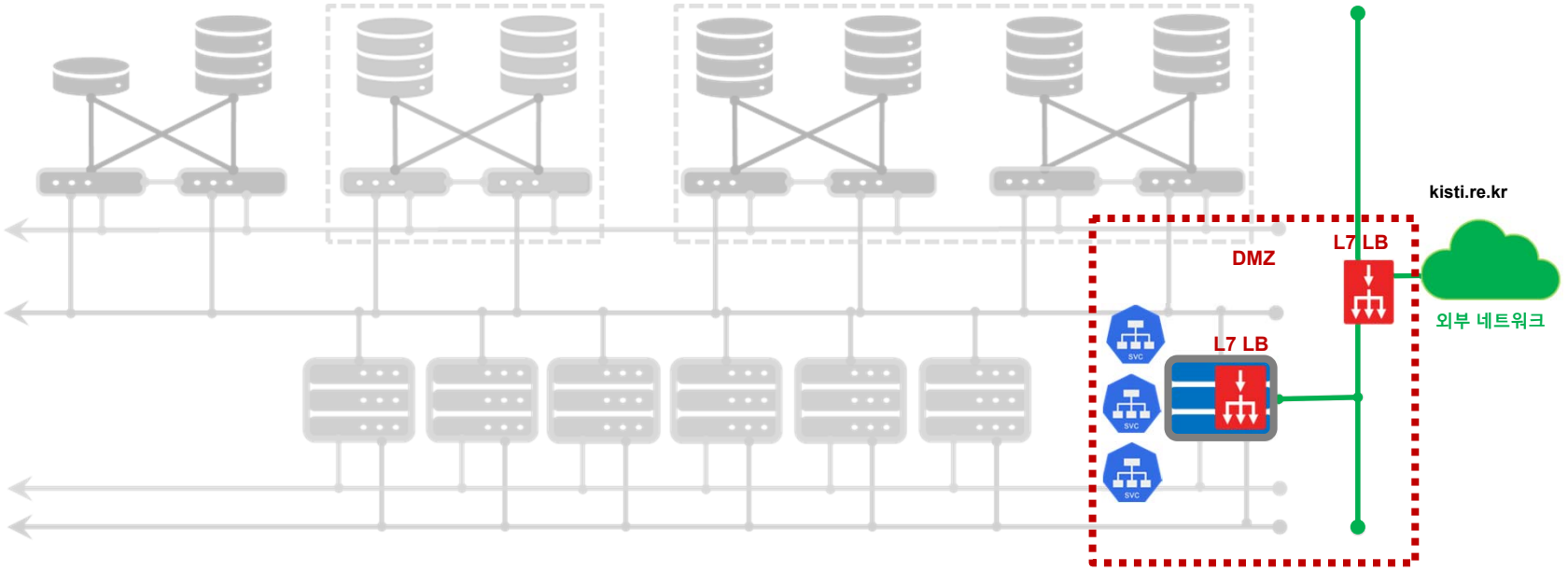
- **DMZ 구성**
  - 클라우드 포털 서비스의 가속 처리와 보안 강화 DMZ 구성을 위한 LB(Load Balancer)와 방화벽
  - 멀티클라우드 수용 서비스 성능 확장 구조 (VPN 연결)



james@jslab.kr

3장. MSA @ Private Cloud

- **L7 LB(Load Balancer) 구성**
  - Hash 기반 정책: IP/Port, Cookie, Protocol
  - 요청 URL 주소를 클라우드 서비스의 지정 IP에 연결
  - L7 LB 제공 위치: Hardware , 서버의 IaaS VM, 서버의 Cloud Native Ingress

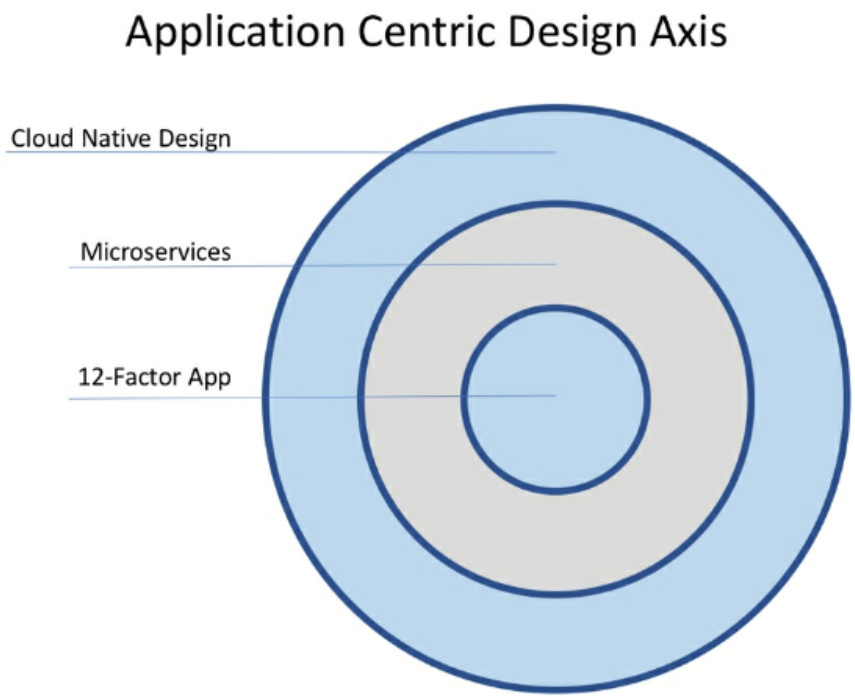


james@jslab.kr

4장. MSA 기술

Application Centric Design Axis

- 12-Factor App
- Microservices
- Cloud Native Design



Microservice ≤ SCS

SCS
(Self-contained system)

- Each SCS is an autonomous web application.
- Each SCS is owned by one team.
- Communication with other SCSs or 3rd party systems is asynchronous wherever possible.
- An SCS can have an optional service API.
- Each SCS must include data and logic.
- An SCS should make its features usable to end-users via its own UI.
- To make SCSs more robust and improve decoupling shared infrastructure can be minimized.

https://scs-architecture.org/#autonomous

james@jslab.kr

4장. MSA 기술

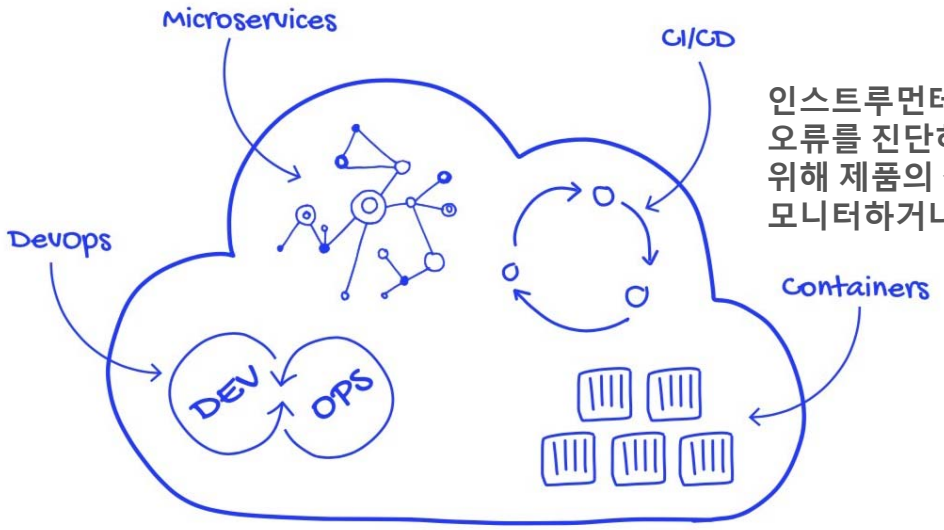
Twelve-factor app design principles

1. **Codebase:** One codebase tracked in revision control, many deploys
2. **Dependencies:** Explicitly declare and isolate dependencies
3. **Config:** Store config in the environment
4. **Backing services:** Treat backing services as attached resources
5. **Build, release, run:** Strictly separate build and run stages
6. **Processes:** Execute the app as one or more stateless processes
7. **Port binding:** Export services via port binding
8. **Concurrency:** Scale out via the process model
9. **Disposability:** Maximize robustness with fast startup and graceful shutdown
10. **Dev/prod parity:** Keep development, staging, and production as similar as possible
11. **Logs:** Treat logs as event streams
12. **Admin processes:** Run admin/management tasks as one-off processes

james@jslab.kr

4장. MSA 기술

- **Cloud native design considerations**
  - **Instrumentation**
  - **Security**
  - **Parallelization**
  - **Resiliency**
  - **Event-driven**
  - **Future-proofed**

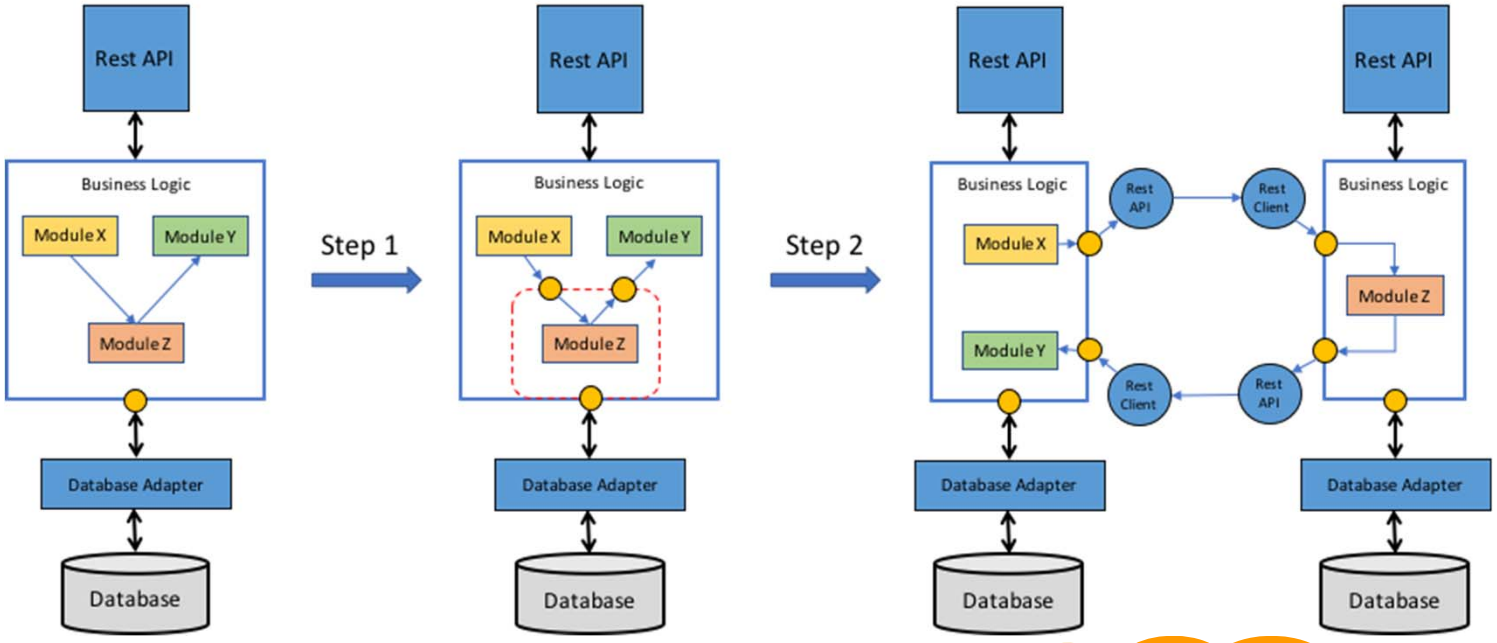


인스트루먼테이션 (instrumentation)은
오류를 진단하거나 추적 정보를 쓰기
위해 제품의 성능 정도를
모니터하거나 측정하는 기능

https://medium.com/walmartlabs/cloud-native-application-architecture-a84ddf378f82

4장. MSA 기술

- **Refactoring** (from Monolithic Module to MSA)
  - **서비스 모듈간 메시지 송수신 필요** (API 사용)

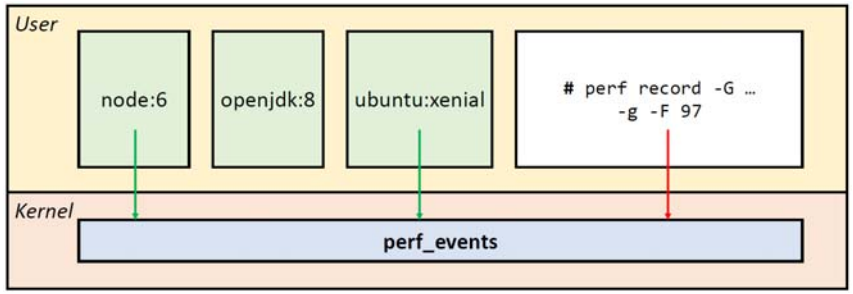


https://haloworld.tistory.com/24 [Halo World:)]

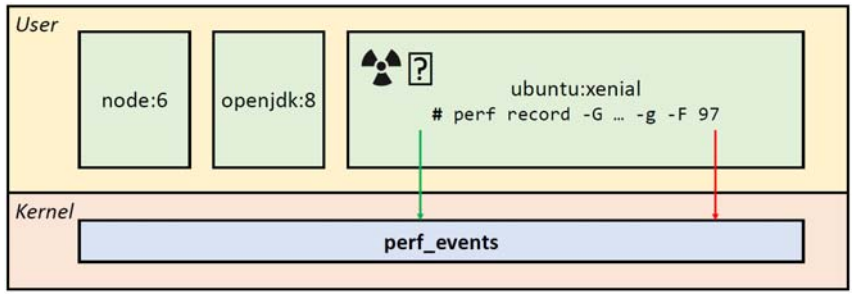
4장. MSA 기술

Tool Deployment

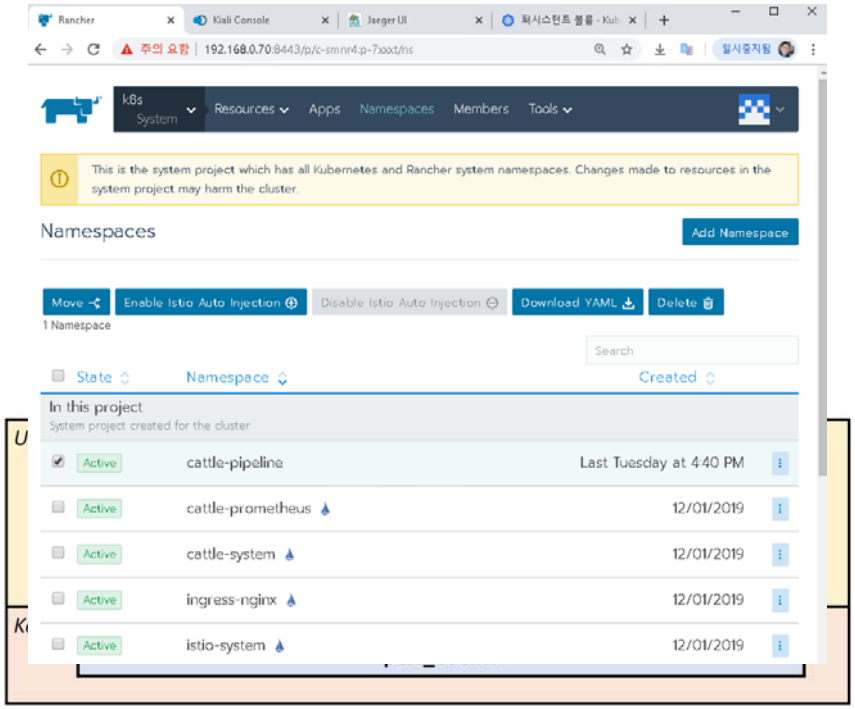
- 호스트 내 설치 (On The Host)
- 컨테이너 내 설치 (In Container)
- 도구 전용 컨테이너 (“Sidecar” Container)



On The Host



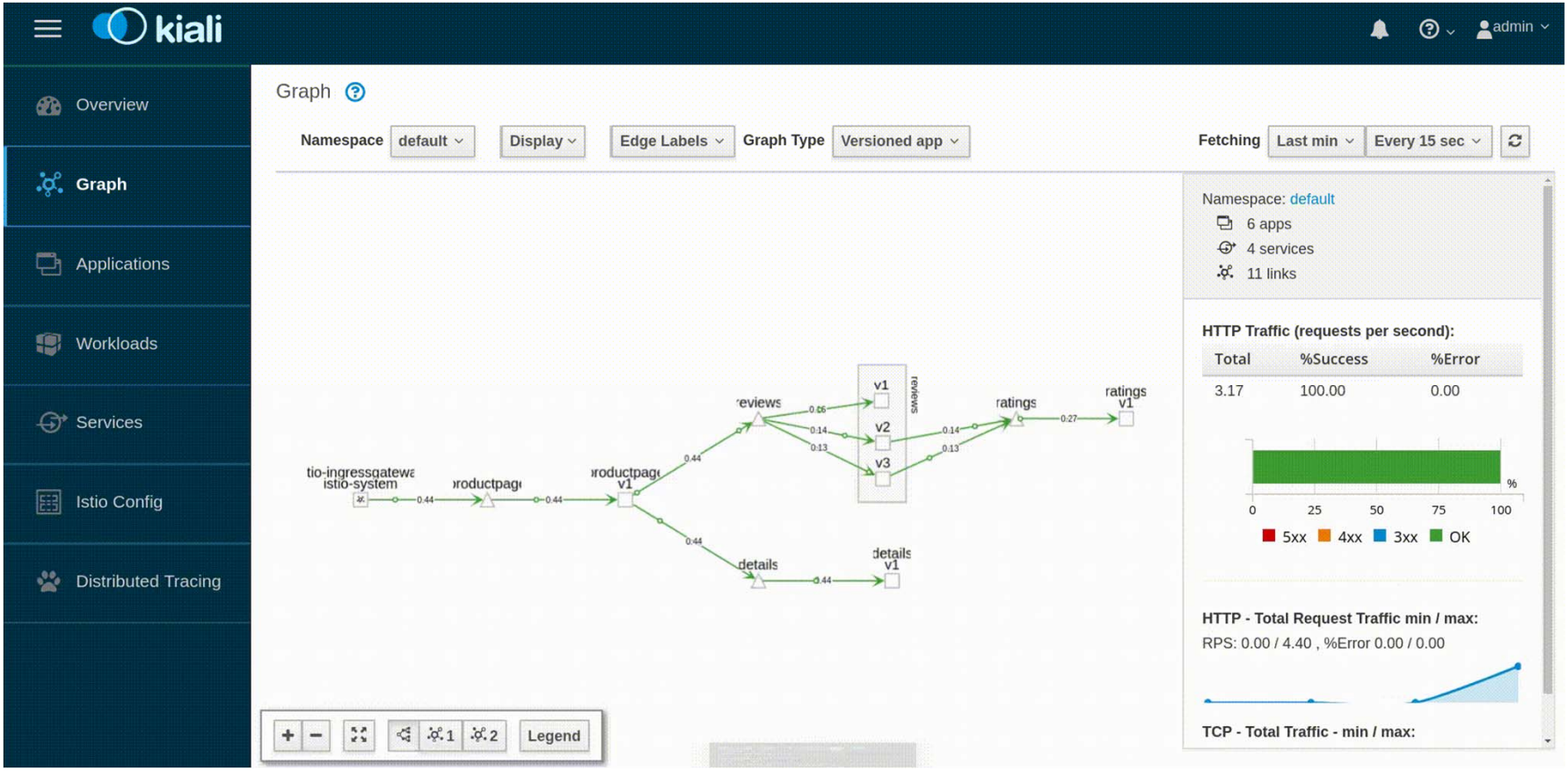
In Container



“Sidecar” Container

4장. MSA 기술

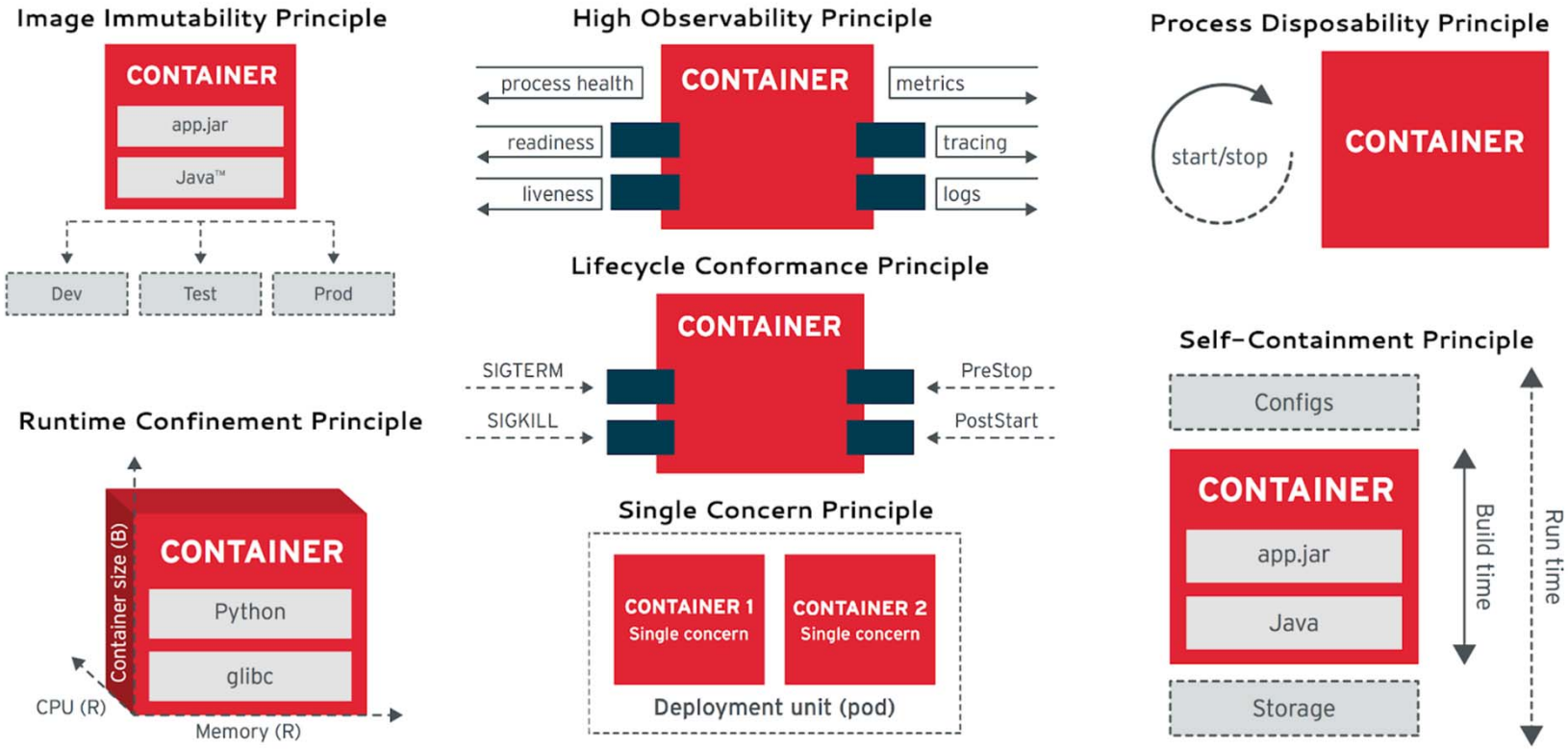
- **Service Mesh 관리**



https://github.com/ruzickap/k8s-istio-demo

4장. MSA 기술

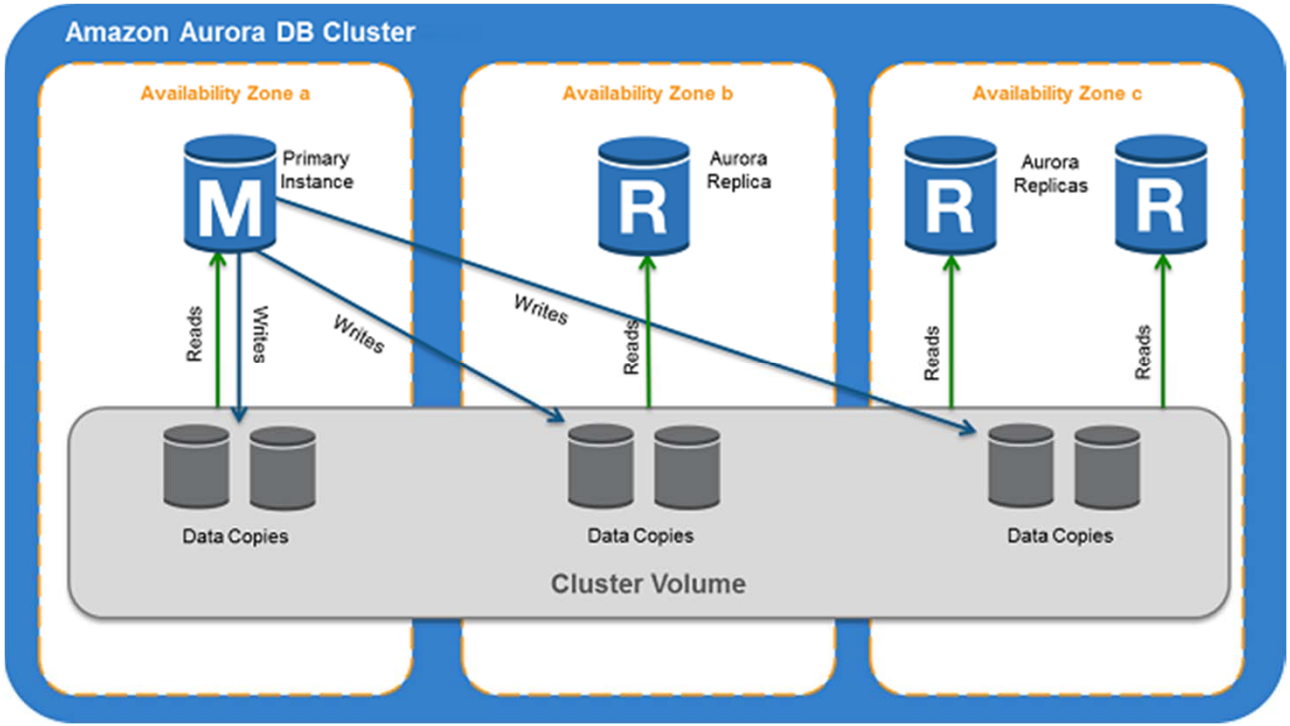
- **Container-based Application Design**



https://kubernetes.io/blog/2018/03/principles-of-container-app-design/?fbclid=IwAR2oMrdP0d1Q6LXebtxNPnt--RS5DIlkCIwpaMSL5mmW7VMaQb6hRV8hkd38

4장. MSA 기술

- **Cloud native managed services (예)**
  - **AWS has a managed database service, Amazon Aurora, built on a fully distributed and self-healing storage service deigned to keep data safe and distributed across multiple availability zones.**



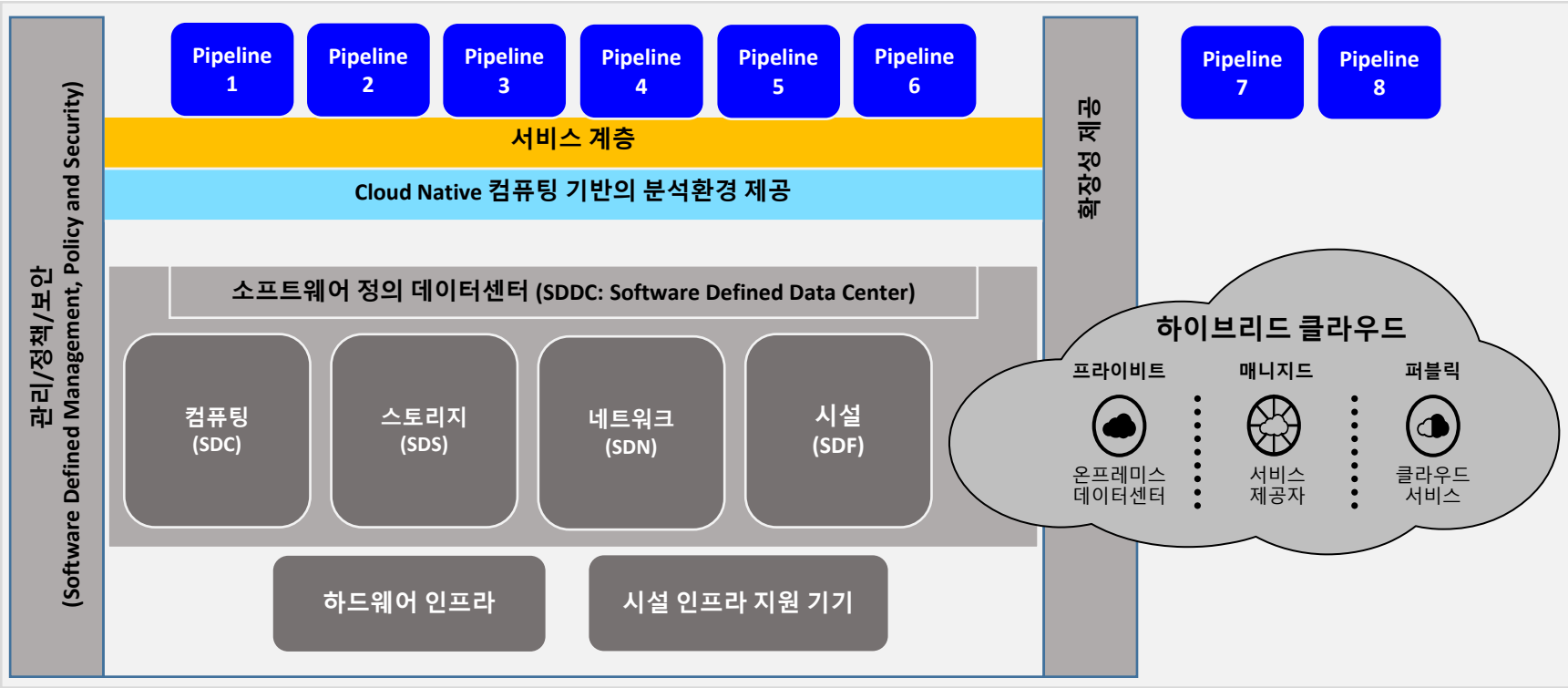
https://docs.aws.amazon.com/AmazonRDS/latest/AuroraUserGuide/Aurora.Overview.html

james@jslab.kr

4장. MSA 기술

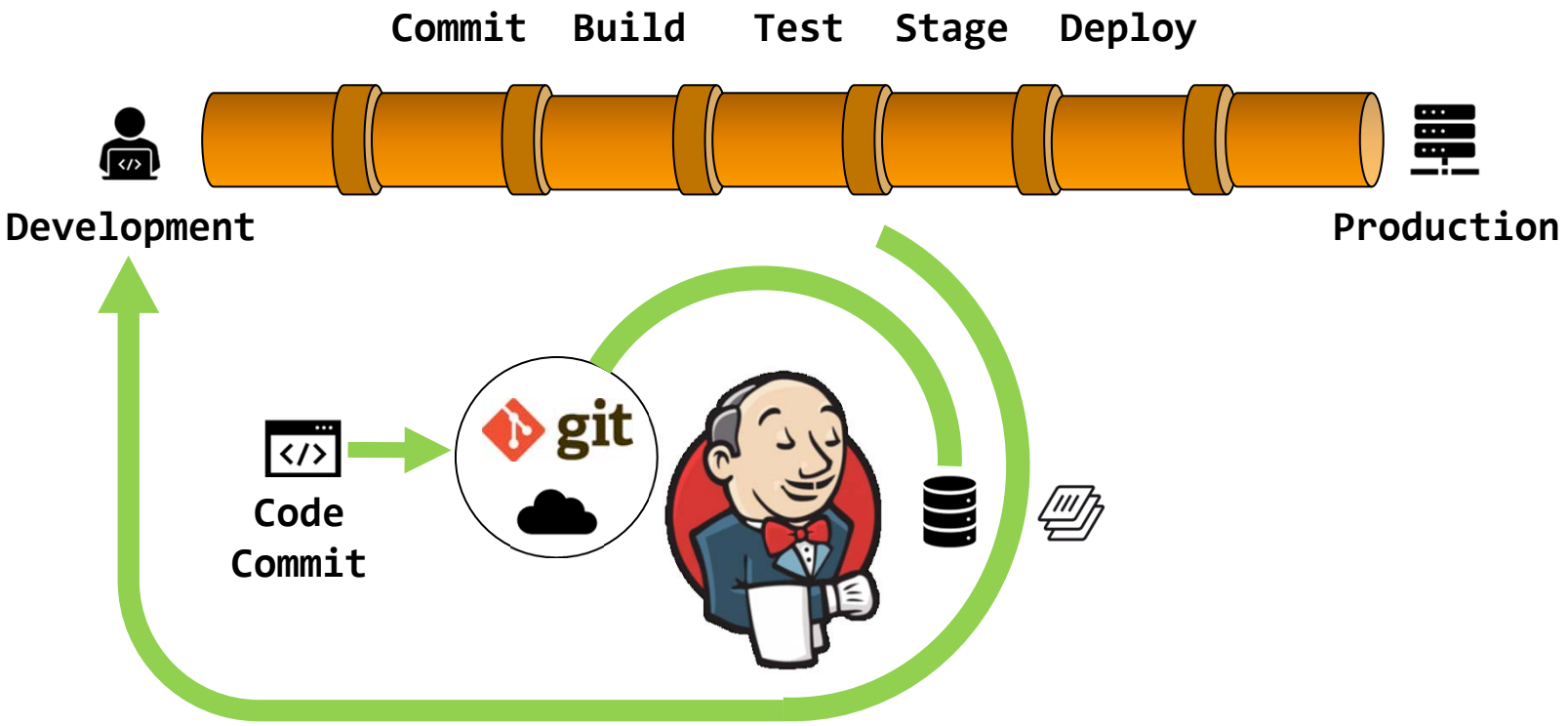
- **서비스의 클라우드 네이티브화** (Cloud Native)
  - **서비스 중단없는 배포, 자가 복구** (Self-healing)
  - **컴퓨팅/스토리지/네트워크/DB/소프트웨어 자원 등의 유연한 사용**
  - **호환성 이슈 해결과 사용의 편리** (User Friendly)
  - **빠른 서비스 제공**

- **SDDC 기반 인프라의 클라우드 수용** (for Enterprise)
  - **Cloud Native등의 기술 발전 수용 SDDC 데이터센터 구축**
  - **데이터센터 자원의 추상화 관리 (SDDC)**



5장. DevOps와 CI/CD

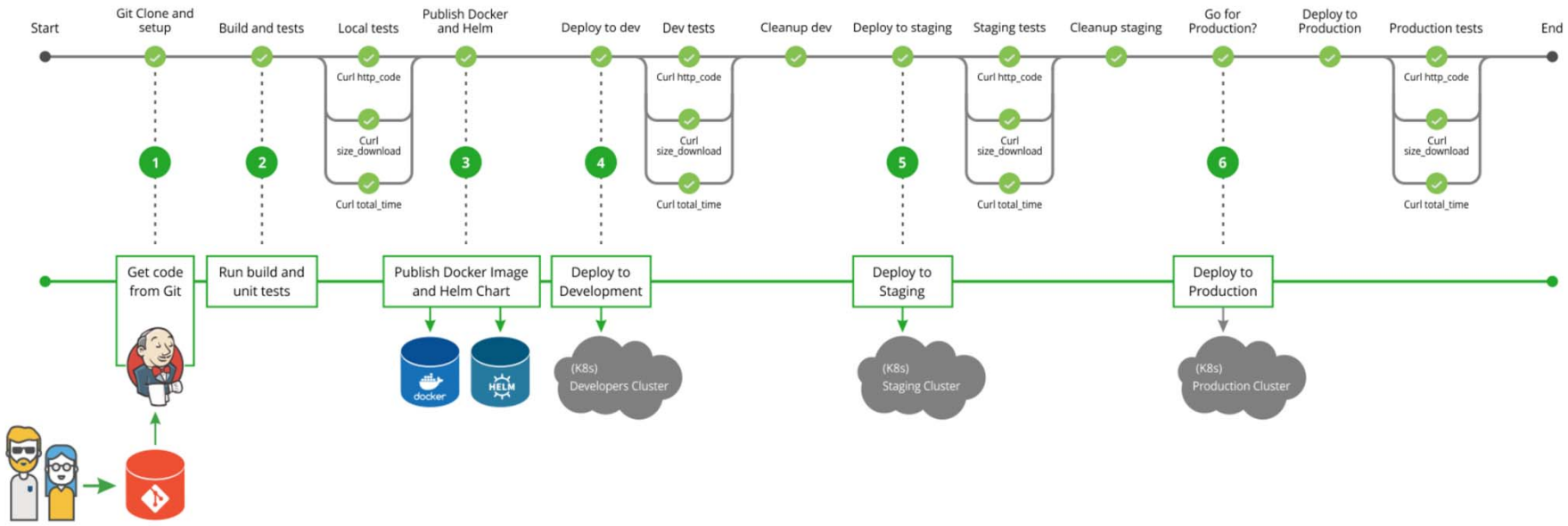
- **Jenkins**
  - **Cross-platform: CI/CD**



https://wiki.jenkins.io/display/JENKINS/Meet+Jenkins

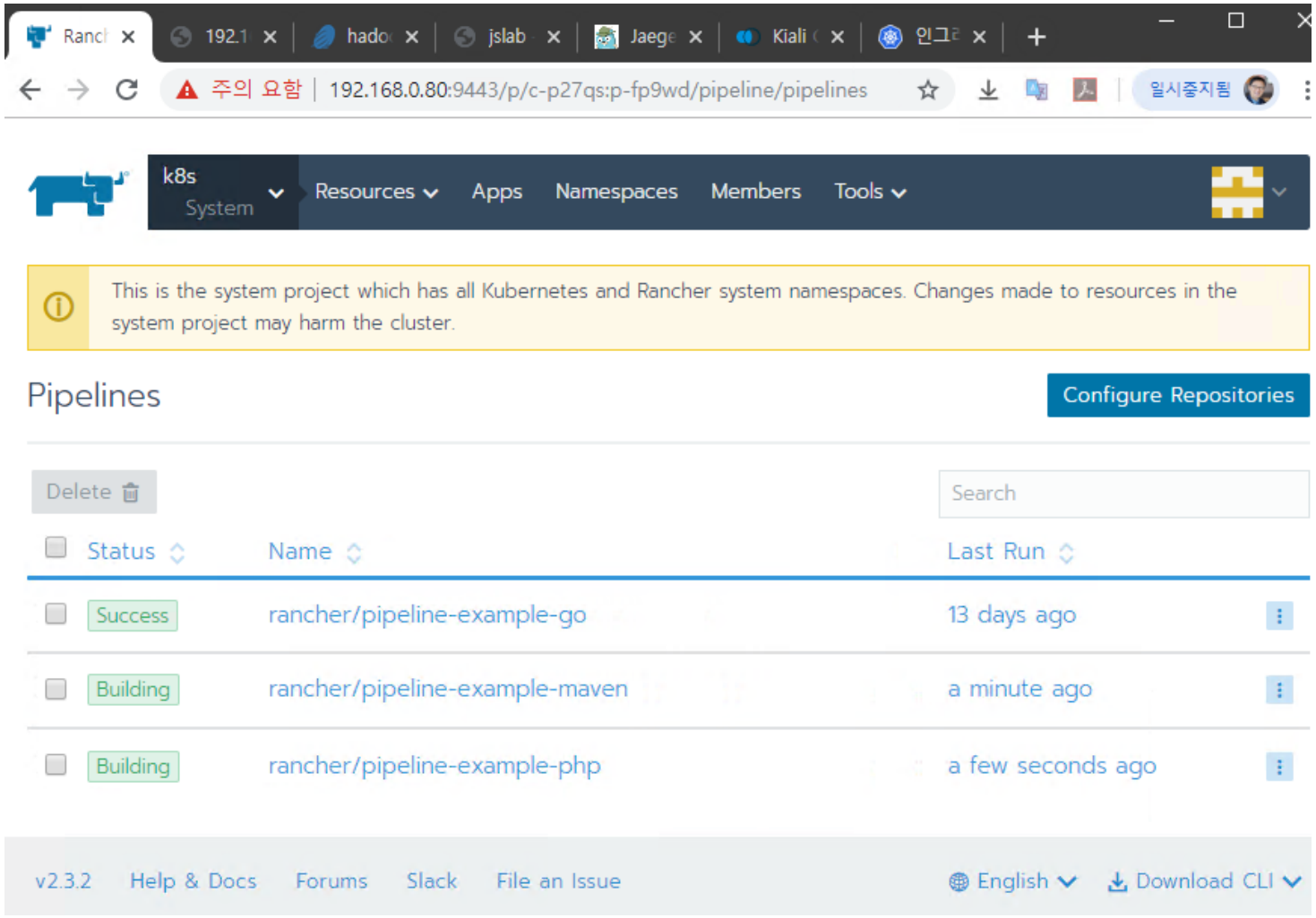
5장. DevOps와 CI/CD

- **CI/CD Tools – Container-based**
  - Immutable Image
  - API Testing
  - Blue/Green, Canary
  - 도구: Jenkins + Docker + Spinnaker + Helm + Kubernetes



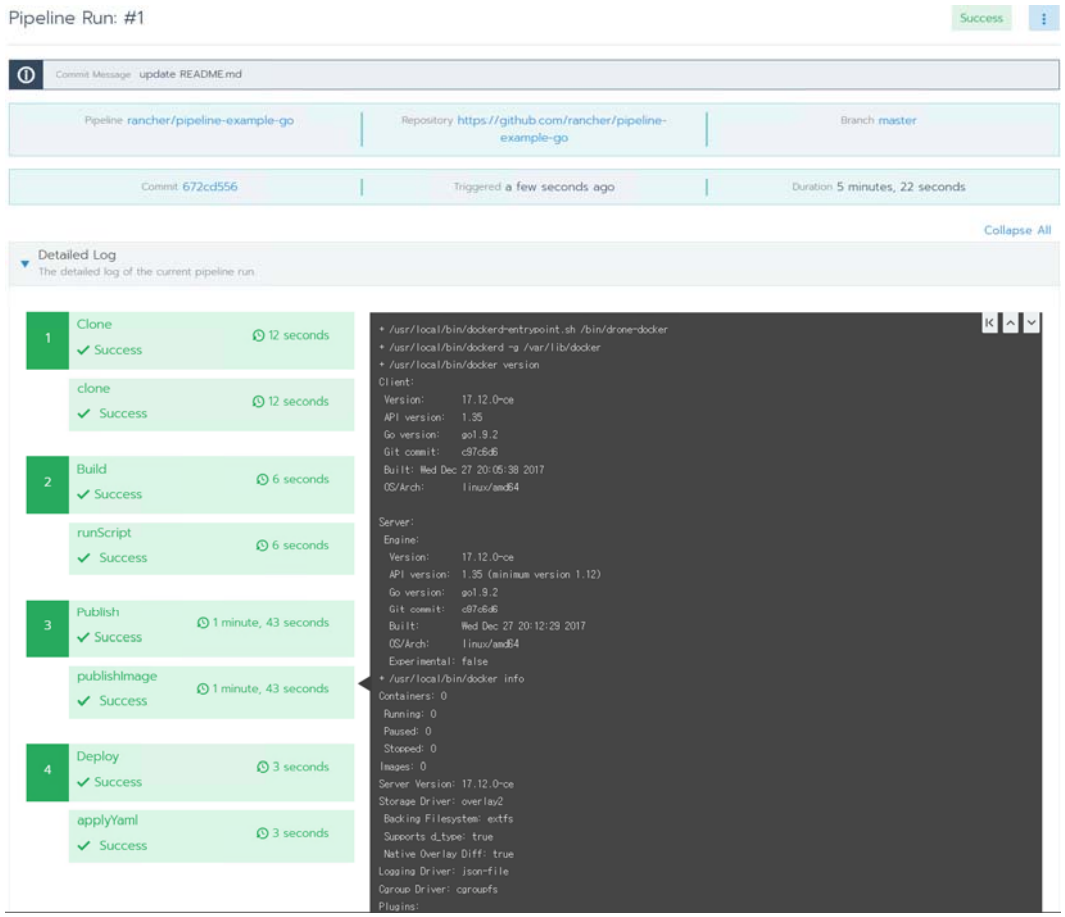
5장. DevOps와 CI/CD

- **Pipeline @ Rancher**



5장. DevOps와 CI/CD

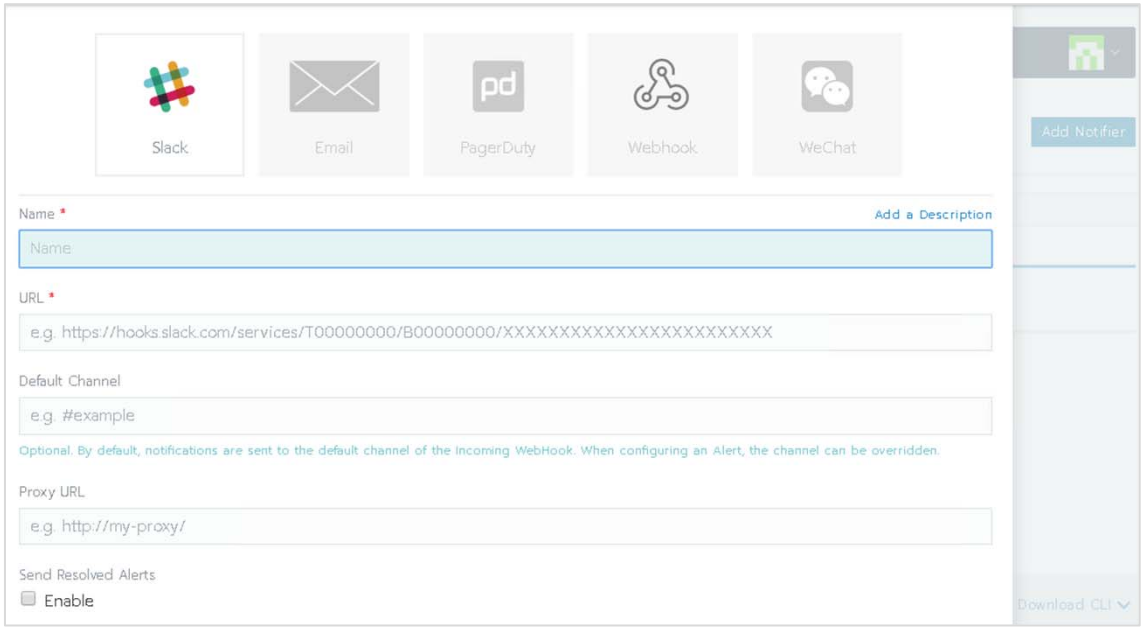
- **Pipeline for K8s**
  - **K8s Pipeline @ Rancher 2.3**



5장. DevOps와 CI/CD

Integrations

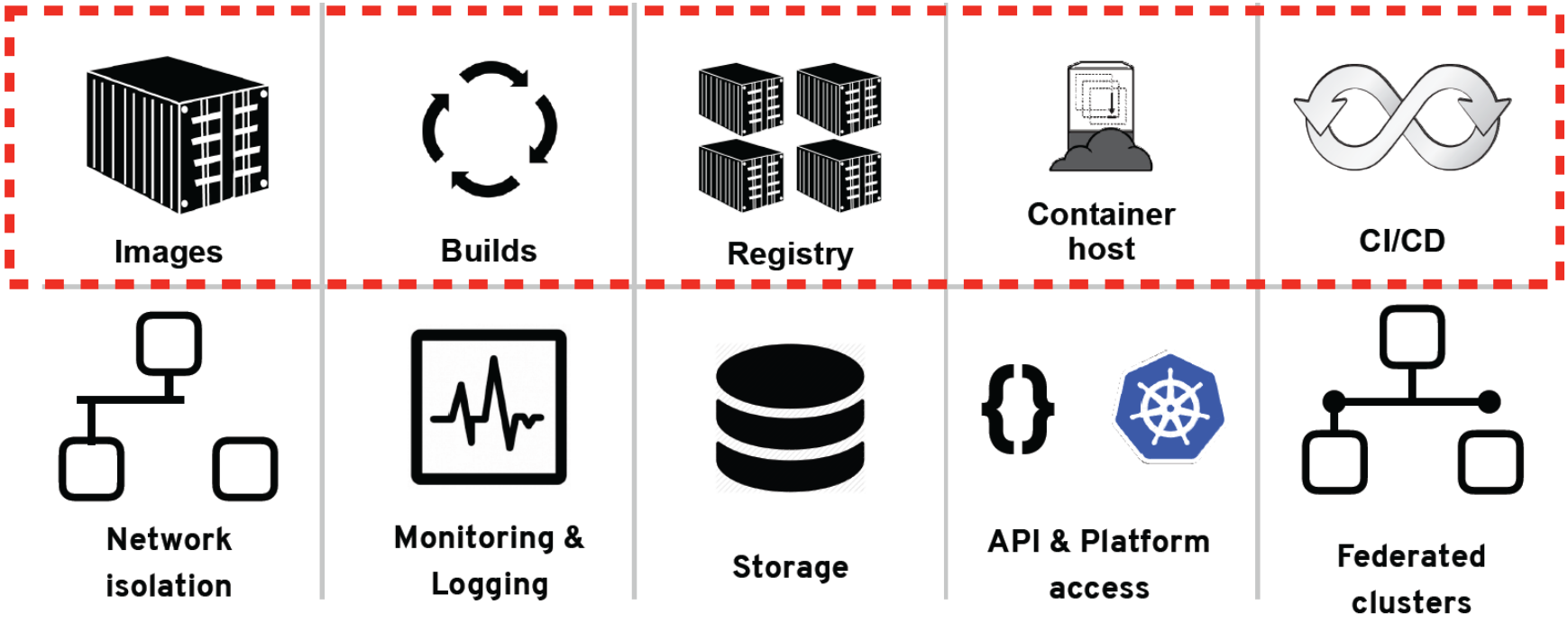
- Shippable integrates well with all popular CI/CD and DevOps tools like GitHub, Bitbucket, Docker Hub, JUnit, Kubernetes, Slack, Terraform, etc.
- For more details about these and other tools please refer to Shippable's website.



james@jslab.kr

5장. DevOps와 CI/CD

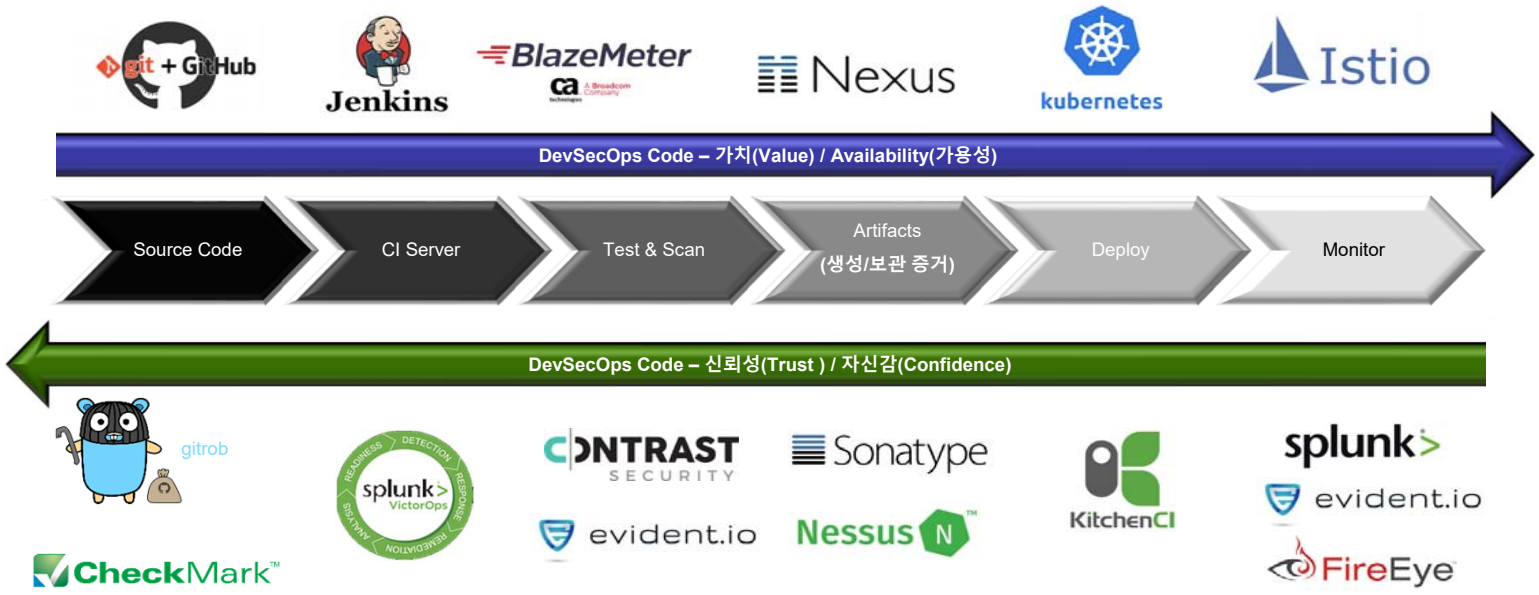
- **CI/CD 환경 보안**
  - 추가 고려: Images, Builds, Registry, Container Host, CI/CD



james@jslab.kr

5장. DevOps와 CI/CD

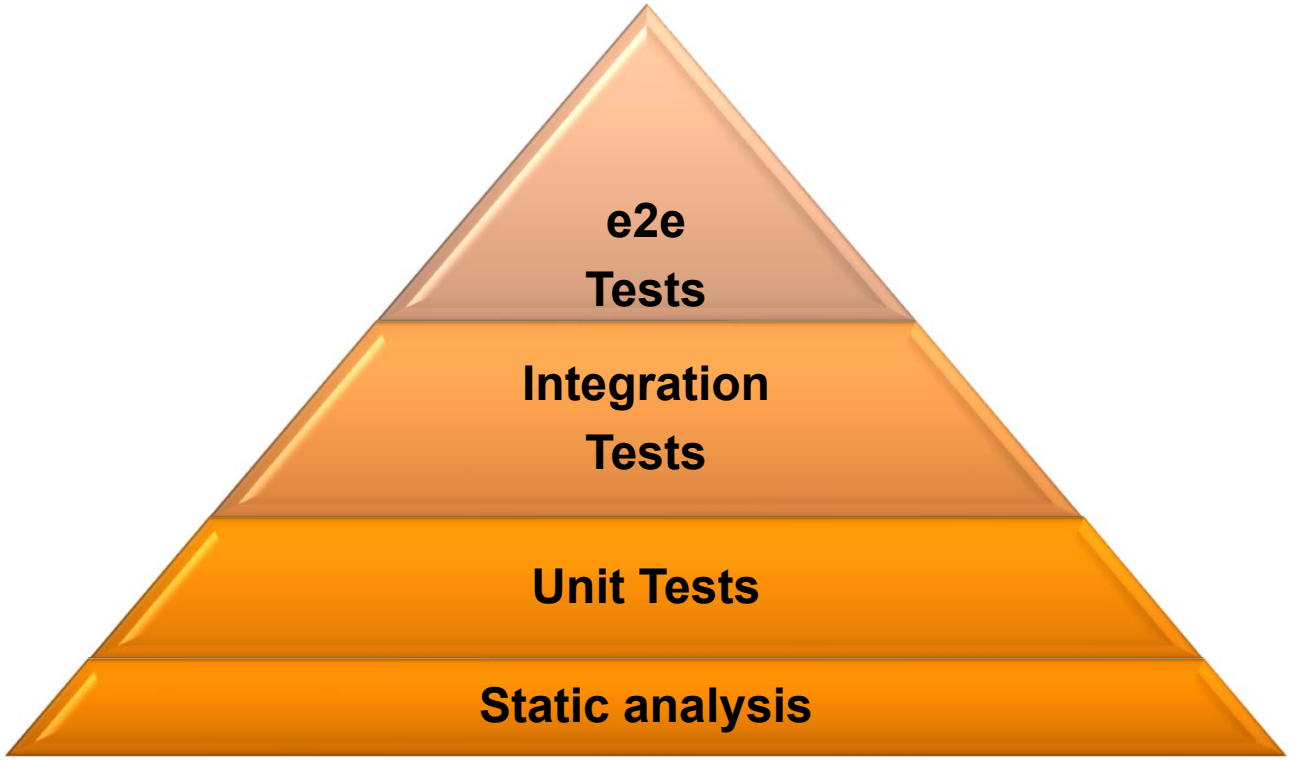
- CNA 기반 단계별 보안 자동화 체계 (DevSecOps)
- Images, Builds, Registry, Container Host, CI/CD 등의 오픈소스 도구를 위한 전문 보안 솔루션 적용



james@jslab.kr

5장. DevOps와 CI/CD

- Test pyramid
  - HTTP requests
  - API calls
  - SSH commands
  - etc.



james@jslab.kr